# SIMULTANEOUS LOCALIZATION, MAPPING AND OBJECT TRACKING IN AN URBAN ENVIRONMENT USING MULTIPLE 2D LASER SCANNERS

By

NICHOLAS MCKINLEY JOHNSON

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

2010

To my family

# ACKNOWLEDGMENTS

First, I would like to thank my family for their love and support during my time at the University of Florida. They have always believed in my potential and have never placed any undue expectations on me. I thank them for allowing me the freedom to pursue my dreams.

I would also like to thank my advisors Dr. A. Antonio Arroyo and Dr. Carl Crane for their guidance and support over the years. It was only through their help that I was able to participate in the 2007 Urban Challenge and work on a myriad of interesting projects during my time at the Center for Intelligent Machines and Robotics. In addition, I would like to thank the other members of my committee, Dr. Douglas Dankel, Dr. Herman Lam, and Dr. Eric Schwartz for their valuable input and guidance over the past few years. This research was made possible in part through the support of the Air Force Research Lab at Tyndall Air Force Base in Panama City, Florida so I would like to thank all the staff that I have had the opportunity to work with out there.

I would like to heartily thank all of my co-workers, and especially my friends, at CIMAR who I have worked with closely while at UF. Although, there were many challenges and many long days and late nights, their support and friendship made each day an enjoyable experience and made it possible to make it through to the end.

Finally, I would like to thank God for making all things possible.

TABLE OF CONTENTS

page

LIST OF TABLES

Table page

LIST OF FIGURES

Figure page

LIST OF ABBREVIATIONS

| | |
|---|---|
| AFRL | Air Force Research Laboratory |
| AS-4 | Aerospace Standard Unmanned Systems Steering Committee |
| CIMAR | Center for Intelligent Machines and Robotics |
| DARPA | Defense Advanced Research Projects Agency |
| DATMO | Detection and Tracking of Moving Objects |
| DGC | DARPA Grand Challenge |
| DUC | DARPA Urban Challenge |
| EKF | Extended Kalman Filter |
| EM | Expectation Maximization |
| EMM | Environment Mapping and Monitoring |
| GPOS | Global Position and Orientation Sensor |
| GPS | Global Positioning System |
| HLP | High Level Planner |
| ICP | Iterative Closest Point |
| IDC | Iterative Dual Correspondence |
| IEPF | Iterative End Point Fit |
| IMRP | Iterative Matching Range Point |
| IMU | Inertial Measurement Unit |
| JAUS | Joint Architecture for Unmanned Systems |
| LADAR | Laser Detection and Ranging |
| LRF | Laser Range Finder |
| LSS | LADAR Smart Sensor |
| LWM | Local World Model |
| MDF | Mission Data File |

| | |
|---|---|
| MO | Moving Object Detection Sensor |
| MPA | Most Probable Angle |
| NFM | North Finding Module |
| RANSAC | Random Sample Consensus |
| RN | Roadway Navigation Motion Planner |
| RNDF | Road Network Definition File |
| ROI | Region of Interest |
| SAE | Society of Automotive Engineers |
| SCRIM | Sampling and Correlation based Range Image Matching |
| SLAM | Simultaneous Localization and Mapping |
| SLAM+DATMO | Simultaneous Localization, Mapping and Moving Object Tracking |
| TSS | Terrain Smart Sensor |
| TTL | Time to Live |
| SSC | Subsystem Commander |
| UMS | Unmanned System |
| WMKS | World Model Knowledge Store |

Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

# SIMULTANEOUS LOCALIZATION, MAPPING AND OBJECT TRACKING IN AN URBAN ENVIRONMENT USING MULTIPLE 2D LASER SCANNERS

By

Nicholas McKinley Johnson

December 2010

Chair: A. Antonio Arroyo
Cochair: Carl Crane, III
Major: Electrical and Computer Engineering

The role of robotics has grown rapidly within the last few years. No longer are they found only on the assembly line but have been tasked with cleaning homes, mowing lawns, and protecting lives on the battlefield. As their responsibilities become more complex, the need for safety grows in importance. Therefore, it is critical that a robot be able to detect and understand elements in the environment. Laser range finders (LADAR) have been popular sensors for use in object detection applications such as Simultaneous Localization and Mapping (SLAM) and the Detection and Tracking of Moving Objects (DATMO) due to their high range accuracy, low cost, and low processing demands. However, these applications have commonly been treated separately despite evidence that they are related. The presence of moving objects adversely affects SLAM systems while static objects are commonly misidentified in DATMO applications. One approach to addressing the shortcomings mentioned has been to combine the two applications in a Simultaneous Localization, Mapping, and Moving Object Tracking (SLAM+DATMO) method. However, past efforts have relied on grid-based approaches which require greater memory and processing power due to the

use of image processing techniques. In addition, no previous work has been attempted to use multiple LADAR to provide a wider field of view which allows the robot to understand more of the world and avoid threats. The work presented here addresses some of the shortcomings described. A novel SLAM+DATMO approach is introduced that represents the detected objects using line segments and polygons, which are more concise and can be processed more quickly. Also, a formal approach for fusing data from two laser range finders is introduced to provide a low cost and simple solution for improving sensing capability. Finally, a mechanism for sharing detected object data to other software components is outlined through the use of a centralized world model knowledge store.

# CHAPTER 1
# INTRODUCTION

Robotics is a rapidly developing field. Initially used primarily on the assembly line since the introduction of Unimate at General Motors in 1961 [1], robots have grown to take on many different roles. Today's robots can be seen, not only on the assembly line but also cleaning people's homes, mowing lawns, and protecting soldiers on the battlefield. One of the areas currently getting a lot of attention is the area of unmanned vehicles. These vehicles are being used to accomplish missions in dangerous situations where human life would be at risk. Many of the unmanned vehicles currently deployed on the battlefield are operated remotely or with human supervision. However, as unmanned vehicles continue to evolve there is an increasing desire to have these vehicles work independent of human interaction. The research presented in this document will hopefully assist in reaching that goal.

The rest of this chapter provides an introduction to the area of unmanned vehicles and some of the general problems that need to be solved. It ends by introducing the specific problem to be addressed and the motivation for the research. Chapter 2 provides a summary of the related literature reviewed in devising the presented work. Chapter 3 introduces the implemented approached and chapter 4 discusses the testing methodology. Experimentation results are given in chapter 5 along with a discussion of the limitations, problems and lessons learned. Finally, chapter 6 proposes some areas of future work and presents the research conclusions.

## Background

Previously, robotics dealt heavily with doing repetitive tasks that were difficult or monotonous for humans. Robots were not expected to make decisions but instead

simply executed a series of instructions. As the field evolved, more responsibility was placed on robots and the demand for them to complete tasks without human interaction grew. The concept of autonomous operation and the field of unmanned systems developed from these demands. The term fully autonomous is defined as “a mode of operation wherein the unmanned system (UMS) is expected to accomplish its mission without human intervention” [2]. When considering a fully autonomous vehicle the level of cognition required is staggering. In general, a robot must be able to reason about a designated mission and devise and execute a plan of action. However, actually building a vehicle to do this is extremely difficult.

One of the most basic missions for any autonomous vehicle is moving from one point to another. However, in order to plan a path through the world, the robot must first have some knowledge about the world. This knowledge is either provided to the robot a priori or is determined through the use of sensors. A robot’s understanding of the world usually begins with localization. A robot must know its starting position in the world before it can plan a path to an end point. When the robot moves it must know its new position in relation to the end point to continue along the plan. One simple and common approach is to use wheel encoders or other odometry data to estimate the robot’s movement from its starting position. Although simple, this method does not give the robot a global sense of the world. Another approach used, especially in indoor environments, is Simultaneous Localization and Mapping (SLAM) [3]. In SLAM, a map of the world is generated as the robot moves and is used to estimate its position relative to the map. However, if the map is incomplete or the environment is very symmetrical the robot cannot be certain of its position. One popular approach for outdoor

applications is the use of the Global Positioning System (GPS). GPS allows a robot to easily and fairly accurately estimate its global position in the world. However, it suffers from problems with noise and errors caused by large structures, such as mountains, buildings, etc.

Localization alone is usually not enough. The world can be a dangerous place and the robot must be able to detect and avoid obstacles and other dangerous situations. Cameras and laser range finders (LADAR) are just two of the many sensors currently used to perceive the world. Cameras provide a lot of information but give inaccurate range readings, and the data output has typically been difficult and slow to process. LADAR, on the other hand, provide very accurate range information, and can usually be processed fairly quickly, but do not provide as much information as cameras. Regardless of the sensor used, it is their responsibility to detect obstacles and help guide the robot safely through the world.

When discussing sensors and object detection, questions about how the objects should be represented arise. In order to avoid obstacles, the robot's path planning element must understand that an object exists and where it is located in the world. A representation of the world, called a world model, which the robot understands is required. There are two general approaches to world modeling: raster based and vector based. In the raster-based approach the world is represented as a series of discrete cells. Each cell stores a value that defines some information about the world, such as occupancy or traversability. On the other hand, the vector-based approach, attempts to model the world as a series of geometric shapes, such as lines, circles, polygons, etc.

Now that the robot has an understanding of the world, knows where it is located in that world, and where are good and bad areas to move in, it can finally plan a path to achieve its goal. There are many options when considering planning algorithms. The chosen algorithm will depend on a number of factors including: the world model, the mission goal, the level of complexity and the level of intelligence. One simple approach may be to drive towards the goal until an object is encountered, then turn, move some random distance, and try again. Another approach treats path planning as a search problem. Every possible path from the current position to the end goal is considered with the best path chosen. Search algorithms, such as the A* and D* algorithms, are used to minimize the number of paths evaluated and decrease processing time. Some heuristic measure, such as travel distance or travel time, is used to choose one path over another.

Finally, the robot must execute its plan and move itself through the world. Movement is accomplished through control commands which adjust the vehicle's state, such as its heading or speed. These commands are executed by low level electro-mechanical systems, such as hydraulic pumps, electric motors or other actuators, which are tied to the vehicle. Actuators attached to the steering column on a ground vehicle, or to the rudder of an aircraft, can be used to change heading, while speed can be adjusted by controlling the input current to electric motors or the throttle position on a combustion engine.

Depending on the level of complexity and uncertainty in the world it may be necessary for a robot to constantly repeat this process. If the environment is static and the sensors can provide a complete picture of the world, the robot only needs to plan

once in order to complete its goal. However, in real-life the world is dynamic and constantly changing, and sensors have limited perception capabilities. As such the robot must constantly adjust its plan to deal with previously unforeseen situations.

The brief summary given here barely covers the issues involved with developing autonomous systems. Robots that can learn from and improve upon previous actions haven't been discussed and introduce a whole new level of complexity. The computational power required to perform the tasks described are considerable, especially for real-time operation. However, recent developments in computer hardware and the continued advancements in the field have brought the idea of a fully autonomous vehicle closer to a reality.

## Focus

As robots begin to work in more real-life environments, their understanding of the world needs to increase. Simply marking areas as bad and good is no longer sufficient, as areas that may currently be good can quickly become bad. Consider the case of a car on a road. If the car is static, then the area in front of it is safe, however, if the car is moving, that same area becomes very dangerous. Robots need to understand the contextual difference between these two situations or they will put themselves and others into harm's way. Contextual information can also help a robot plan a better route through a city or anticipate possible areas of congestion. If a robot has a map which provides data such as restaurants or schools it may consider the current time and avoid those areas. Two major requirements to developing contextual awareness are better world modeling and perception.

A coherent method for representing and storing objects in the environment which allows for the addition of contextual information is required. Raster based world models

do not allow for attributes, such as object classifications, to be attached to the objects they represent, and as such a vector based world model is required. In order to store the generated model and retrieve it at a later time a World Model Knowledge Store (WMKS) is usually employed. It provides a centralized storage location and allows for multiple elements to access the same stored data. Shared data access is an important capability as robot functions are becoming more distributed as they evolve, and information exchange becomes critical

Robots also need to detect, track, characterize, and differentiate between static and dynamic objects in the environment, such as cars and buildings, or people and trees. One of the many questions that come up is how to isolate static and dynamic objects in the environment. Previously, a lot of research considered them separately and dealt with either one or the other. It is only recently that both static and dynamic objects have been considered together. Once static and dynamic objects have been isolated from each other, it makes sense to map the static objects to aid in localization and route planning. The use of GPS for localization is highly error-prone and although, much work has been done in combining GPS with other sensors to improve performance, it can rarely provide the level of accuracy that is needed to function safely in a city environment. The use of the generated map and the application of SLAM techniques can greatly improve localization.

When discussing perception, sensor range and viewable area is critical. Ideally, robots need to have a complete view of their surroundings. However, very few sensors can provide a 360 degree coverage area, and those that do are currently very expensive or computationally intensive. The most popular approach to achieve a wider

sensor range has traditionally been the use of multiple overlapping sensors. However, outside of the use of multiple cameras, very little research has been done on how to best combine the data from these multiple sources, especially when building a vector-based world model.

## Problem Statement

A number of interesting questions and issues arise following the discussion above. When dealing with map generation and object tracking, the interaction between static and dynamic objects in real world environments compound the inherent sensor problems, such as noise and occlusion. Also, dynamic objects may be temporarily static when initially mapped and may not be present when the robot returns to a previously mapped area. Multiple sensors provide greater coverage but questions about data fusion and computational load come up. More sensor data increases complexity and could increase processing time. Latency issues, inherent in the use of a WMKS, introduce delay between when an object is sent to be stored and is actually stored. However, sensor processing cannot always wait until storage is complete due to the real-time requirements of an unmanned vehicle. The proposed research addresses some of the issues surrounding localization, map generation, and object tracking in dynamic environments, using multiple laser range finders, and utilizing a World Model Knowledge Store.

## Motivation

The Center for Intelligent Machines and Robotics (CIMAR) was founded in the 1970’s, and since then has made significant contributions in the areas of mechanisms, autonomous vehicles, and intelligent machines. They have participated in all three of

the Defense Advanced Research Projects Agency (DARPA) sponsored robotic challenges and have made the trip to the National Qualification Event (NQE) every time.

While the 2004 and 2005 DARPA Grand Challenges (DGCs) focused on navigation in static, off-road environments, the 2007 DARPA Urban Challenge (DUC) focused on an urban setting, and included the presence of both manned and unmanned vehicles. The robots were required to exhibit a long list of driving behaviors that humans normally perform. Tasks ranging from the basics of maintaining speed limits, executing u-turns, and observing intersection precedence, to advanced maneuvers such as traversing obstacle fields, parking, and merging, were required to be performed [4]. CIMAR, building off of the experience from the previous challenges, developed the Urban NaviGator (Figure 1-1); a 2007 Toyota Highlander Hybrid. In order to be successful, object detection and tracking were critical. The approach chosen was to split the object detection and tracking elements into two separate software entities (components).

The Moving Object Detection Sensor (MO) was responsible for detecting and tracking objects over short periods. The four planar LADAR shown in Figure 1-2 were used for the detection and tracking system. Objects were independently detected from each LADAR scan and matched using a simple centroid and point distance method, with the points from matching objects being combined to form the final new object. The objects from the current scan were matched with previous objects using the same simple centroid and point distance method. The system did not differentiate between static and moving objects, but treated all objects of a given size the same, with static objects having zero velocity. If an object was not detected in the current scan, due to

sensor noise, occlusion or some other reason, that object would not be reported to the rest of the system.

The Local World Model (LWM) was responsible for tracking objects over a longer period of time. It attempted to compensate for occlusion by remembering every object for some defined period. The LWM also assisted in the high level decision-making process by evaluating the future threat level of each object when compared to the known road network. For example, an object detected on the road in front of the robot moving at a slower speed will eventually become a danger. As such, the LWM would recommend a vehicle speed that prevented a collision with the object ahead. The LWM relied heavily on the object ID numbers generated by MO to correlate objects over time. If MO switched the IDs of two objects or lost track of an object for a long period, the higher level decision-making in the LWM would make bad decisions.

CIMAR has also collaborated with the Air Force Research Lab (AFRL) at Tyndall Air Force Base for many years. During that time they have successfully developed vehicles for applications such as clearing mine fields, detecting unexploded ordinance, and patrolling base perimeters. At the end of 2008, AFRL tasked CIMAR with developing an Environmental Mapping and Monitoring (EMM) system. This system required a vehicle that could perform two independent but related tasks, while autonomously navigating or patrolling an area. First, it must be able to generate a human understandable and modifiable map that can be stored and later retrieved, and second, if a map is provided, the system must be able to detect differences between the map and the current environment. Figure 1-3 shows an example scenario for the EMM system.

In the 2007 DUC, a robust and reliable object detection system was required. However, the implemented system was not very robust and the approach of treating all objects in a generic manner did not fare well when dealing with a mixture of both static and dynamic objects. The use of multiple LADAR to enhance the robot's range of perception was not handled well and the short term storage limitations of the system affected decision making in the presence of occlusion. The research presented here seeks to build a more robust system that can deal with both static and dynamic objects, while exploiting the added benefits of multiple LADAR, by addressing the problems and lessons learned in the DUC. It also seeks to meet the requirements for the EMM system outlined by AFRL. A more in depth discussion and analysis of the approaches used and problems encountered is given in the Previous Work section.


Figure 1-1. The Urban NaviGator: Team Gator Nation's 2007 DARPA Urban Challenge vehicle.

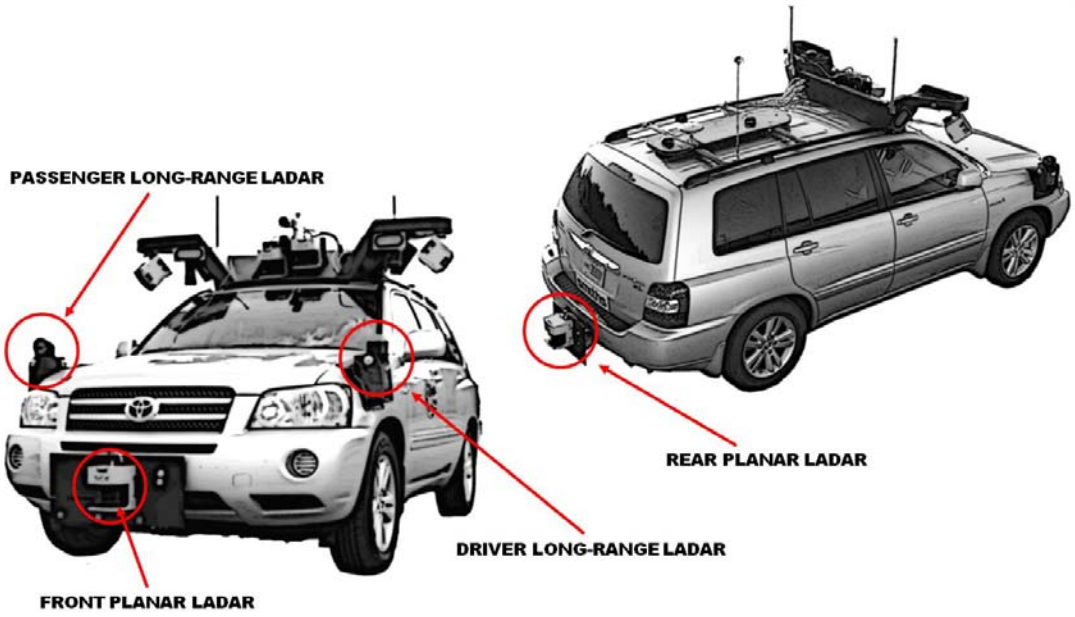



Figure 1-2. Placement of the LADAR used for Object Detection

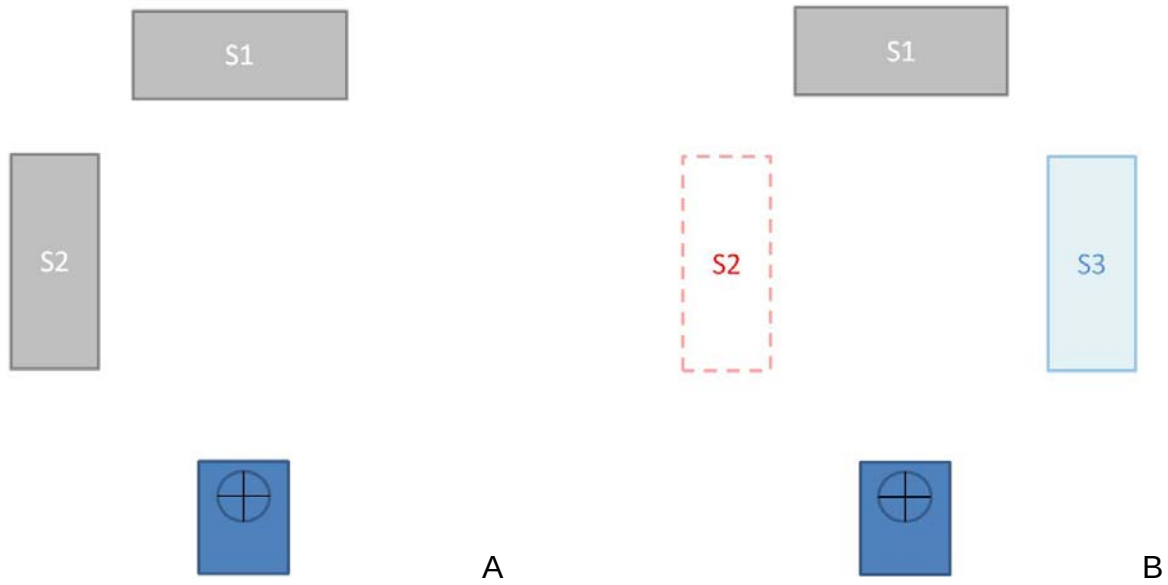



Figure 1-3. EMM example scenario. The robot is represented by the dark blue box. A) The robot generates and stores a map with two objects, S1 and S2. B) The robot returns to a previously mapped area and detects the new object S3 and discovers that object S2 is missing.

# CHAPTER 2
# REVIEW OF LITERATURE

A review of published research was conducted to get an idea of how the problems present in localization, mapping, and object tracking have been approached. Although a lot of work has been done using cameras, the focus was placed on papers dealing with the use of 2D LADAR. The review began with an investigation into previous SLAM work, which has been studied extensively. However, many of the approaches only consider static, indoor environments and do not deal with moving objects. Therefore, papers discussing the Detection and Tracking of Moving Objects (DATMO) where reviewed next. A wide range of work has been done in this area, from tracking pedestrians in indoor environments to tracking cars and bicycles in urban environments. However, most of the work done treat moving objects in isolation and do not consider the stationary objects in the world. Objects of interest are filtered based on size and geometry constraints or based on their location in the environment. Stationary objects that cannot be filtered adversely affect the developed algorithms. Finally, work done in Simultaneous Localization, Mapping and Moving Object Tracking (SLAM+DATMO) was reviewed. The area of SLAM+DATMO is still relatively new and there are many problems still to be tackled. One fundamental issue is how to detect moving objects and separate them from static objects in the world. The review process revealed many techniques for dealing with LADAR data that were applied in all three domains. These techniques will be discussed below.

## Simultaneous Localization and Mapping

SLAM is the process by which a robot builds a map of the environment and uses that map to deduce its location. In most SLAM applications an initial position estimate is

known using odometry or some other mechanism, which is corrected using the currently sensed environment and the stored map. The “loop closing” problem is one of the biggest and most important problems in SLAM and deals with correctly recognizing previously mapped areas and re-associating them to the currently sensed environment [3]. The general steps required for SLAM are outlined in Figure 2-1. It can be seen that there are two main steps: environment representation and map association. There are three general approaches to environment representation: grid-based, feature-based and direct methods. Each of the three approaches along with a novel hierarchical approach is discussed below. The chosen representation dictates the method used in the map association stage. One problem with the current SLAM approaches is that the generated map does not treat each real object in the world as a single entity. A single building is represented as a series of grid cells, features, or points which are not related to each other.

**Grid-Based Approaches**

Grid-based approaches represent the world as a series of cells, sometimes called an occupancy grid [5] or traversability grid [6]. Each cell in the grid represents a square area of the world, for example in [6] each cell represents a 0.5m square area, and is assigned a value based on if a LADAR strike falls within that cell or not. There are many techniques that are used to determine the value of the cell. In a simple, binary occupancy grid, the cell is either occupied or free, while in a more complex grid, the value assigned represents the probability of occupancy. In grid-based approaches sensor noise can be compensated for, by choosing an appropriate cell size, and map generation is simple. Also, any object can be represented and sensor fusion is straight forward, as multiple sensors can be used to adjust the values in a cell. However,

localization using the grid based approach is difficult [7] [8] and no information about the objects represented is known, whether an object is a tree, building or parked car. Image processing techniques must be applied to extract any further understanding of the environment [8] and to correlate grid maps. Furthermore, they cannot be used to solve the loop closing problem [7]. For a fairly accurate map, a large amount of data must be stored, which increases processing time and inhibits their use for representing large areas of the world.

**Direct Methods**

Direct methods represent the world using raw point data without simplifying or extracting features from the data. Data association is usually performed using the Iterative Closest Point (ICP) algorithm or some variant. In its simplest form, this algorithm calculates the distances between all points and associates the two points with the shortest distance [9]. After all the points are associated, the robot's position estimate is updated and the process is repeated until some convergence criterion is met. In order to associate the points between time frames, all the points are transformed to a common reference frame. In [9] an occlusion hypothesis is applied that detects and removes occluded points in order to minimize the association error.

One problem with the general ICP algorithm is that associated points have different rotational displacements. Ideally, every point undergoes the same rotational and translational displacement and, therefore, the ICP algorithm introduces an inherent association error. A variant of the ICP, called the Iterative Matching Range Point (IMRP) algorithm [9], associates points that have the closest range values, given an angular change window, which biases it towards the rotational displacement. It was shown that the translation residuals converge faster in the ICP algorithm while the rotation residuals

converge faster in the IMRP algorithm. Therefore, the ICP and IMRP algorithms are combined in the Iterative Dual Correspondence (IDC) algorithm [9], in order to exploit their individual advantages and produce a more reliable position estimate. The work done in [10] also attempts to apply a uniform rotation displacement to all points by calculating the Most Probable Angle (MPA). A probability for each rotational displacement is calculated and the angle with the highest probability is applied to all the points.

As previously mentioned, it is often assumed that every point undergoes the same rotational and translational displacement and that points between scans always perfectly match. However, sensors are not perfect and introduce some level of uncertainty in their readings. The work in [11] considers the sensor uncertainty and introduces three error values: measurement noise error, measurement bias error, and correspondence error. The measurement noise and bias errors are inherent to the sensor while the correspondence error is a combination of the sensor and position errors. The final matching error is the sum of these three error sources. Therefore, for each point association a correspondence error is calculated which contributes a different weighted value to the matching error. The points that minimize the matching error are associated and a position estimate is generated. The process is repeated until some convergence criterion is met.

The direct method approach is simple and although not discussed in any of the papers, fusion between multiple LADAR would be straight-forward, as all the points are independent of each other and can be stored together. However, this approach requires a lot of memory and processing power in order to store and associate points in a large

or cluttered environment. Also, as with grid-based approaches, direct methods do not provide a mechanism for object understanding. Points are treated independently and are not grouped in any way to represent an object (such as buildings, fences, trees, etc).

**Feature-Based Approaches**

Feature or landmark based approaches compress raw data into a set of pre-defined features, which are chosen based on the robot's operating environment. Line segments are a popular feature for indoor and structured outdoor environments, as they generally tend to have many straight edges that can be easily represented. There are many line extraction techniques and [12] compares three of the most popular: Successive Edge Following, Line Tracking, and Iterative End Point Fit (IEPF). Experimental results showed that the IEPF generated better representations of the world (using a visual determination by a person) across all three tested environments, for a wider range of threshold values than the other two. An IEPF variant called Split and Merge and the Line Tracking algorithm are compared in [13] against a number of other algorithms, such as the Hough transform, the Expectation Maximization (EM) algorithm, and the Random Sample Consensus (RANSAC) algorithm. The paper concluded that the split and merge and line tracking algorithms are preferred for SLAM with the former being the best choice for real-time applications.

Circles are another feature sometimes seen in SLAM applications. They are used in [14] to represent tree trunks and tree-like objects, such as pillars, which are present in semi-structured, outdoor environments and two algorithms are introduced for the extraction of both edges (line segments) and circles. The first method uses an Extended Kalman Filter (EKF) to cluster the points together by estimating the position of the next

point and completing a cluster if the error between the estimate and the actual position is outside a threshold. A line is first fit to the points and the error of the line calculated. If the error is greater than some threshold then a circle is fit using a Gaussian-Newton method. The second method extracts features without clustering and begins by assuming that a circle feature will be extracted. The circle parameters (radius and center point) are initialized, using the first three measurements, and if the radius is below a threshold the circle model assumption is kept, otherwise a line model is used. In either case, an EKF is used to track the points until a discontinuity is detected.

Numerous methods for position estimation were found when dealing with feature-based SLAM. In [15] circles are extracted using a Gaussian-Newton method [14] and matched using a nearest neighbor criterion where the circles that have the closest center positions are associated. A particle filter is used to estimate the robot's position between time frames and is compared against the validated estimates generated by the feature associations. The validated estimate closest to the predicted estimate is used to update the vehicle state. An EKF is used in [16] but only incorporates the features and not the odometry parameters in the position estimate, while [17] uses an information filter. A data association and position estimation technique based on the Possibilistic C-Means algorithm (PCM) is introduced in [18]. Line and ellipsoidal features are extracted and the distance between features over multiple time frames are calculated and minimized to determine feature association. An alternate optimization approach similar to the PCM algorithm is used to update the vehicle position estimate. An ICP based approach which is extended using an EKF (ICP-EKF) is implemented in [19]. A polyline map is generated from the scan data and scan points are associated to the line

segments using an ICP approach. Once the ICP has converged, the EKF is used to further improve the position estimate. Finally, [20] uses a rule-based approach to match features and a non-linear least squares method to estimate the vehicle position. In the approach developed, the chosen features are point clusters which are found using an IEPF based methodology, and matched based on their distance and length relative to each other. Once the clusters have been matched, the non-linear least squares algorithm is applied to generate the final position estimate.

Two of the papers reviewed specifically dealt with map building using features without extending the work to also include localization. These papers introduced a few novel ideas and are worth mentioning. An approach that considers the point uncertainty when extracting line segments is discussed in [21]. Points are grouped using a Hough transform and a line is fit to the points using a maximum likelihood approach, where each point is weighted based on its uncertainty due to sensor noise. The generated line has an uncertainty attached which is used during merging. Lines are merged by converting them into a common reference frame and applying a chi-squared test. If two lines are within a 3 sigma deviance threshold from the combined line uncertainty the lines are merged using a maximum likelihood formulation. A method that matches geometric primitives of line segments and circles for map building is discussed in [7]. A line matching method involving a chi-squared distribution test is utilized and matching lines are fused using a static Kalman filter. Circles are matched using a simple distance criterion and are fused by averaging the two circles. Another novel concept introduced is the idea of a wipe triangle which is used to filter out noisy line segments. When a new line is constructed every line in the map is checked for intersection with the wipe

triangle. If a line intersects with or is inside of the wipe triangle region it is removed or fragmented.

Feature-based approaches to mapping provide a method for compressing the data and therefore, do not need as much storage space as grid-based or direct methods. However, if bad features are chosen, a large error is introduced into the map and the position estimate is affected. One method that has been proposed is the use of artificial landmarks which can help alleviate problems with feature extraction, as the environment is modified to add features that are well known [17] [19]. However, the infrastructure changes that are required to use artificial landmarks make this approach infeasible when considering real-life situations. Although feature-based approaches provide a better understanding of the world than both grid-based and direct methods, most techniques still cause single objects to be decomposed into multiple features without any connection between them. Most feature-based SLAM approaches reviewed only used a single LADAR and did not fuse data from other sensors. Only [16] discussed a method for fusing data between a LADAR and sonar by exploiting the advantages of each sensor and using a Kalman filter. However, it did not discuss any approaches for combining multiple of the same sensor types in order to provide a wider sensor coverage area.

**Hierarchical Object Based Approach**

A novel approach taken by [5] is to combine all three representations in a hierarchical object based approach in order to overcome the shortcomings of each individual approach. LADAR scan points are clustered using a simple distance criterion and clusters from different time frames are associated with each other using the ICP algorithm. Associated clusters are grouped together to form objects and a grid-based

methodology is used to calculate the uncertainty of the cluster associations. The generated grids are stored as features which can be matched using feature-based approaches in order to solve the problem of loop closing. The combination of direct methods and grid-based approaches provide localization within a local frame (close to the vehicle), while the use of the feature-based approach provides localization within a global frame (where the vehicle is located in the map).

The discussed hierarchical approach still suffers from some of the shortcomings of the individual approaches. The use of grid maps as a feature requires a large amount of memory, especially when considering large maps. Although the objects developed are groupings of points that probably belong to the same object, they are not treated in a manner that allows for understanding what the object represents. Finally, data from the grid map features cannot be easily extracted once they have been stored and the raw cluster data has been lost. Ideally, object data should be retrievable when the robot returns to a previously visited area.

### Detection and Tracking of Moving Objects

The reviewed SLAM approaches break down in the presence of moving objects, since they introduce error in the matching process. Therefore, a number of papers were reviewed that specifically examined the DATMO problem. A generalized approach is outlined in Figure 2-2 and is broken up into four mains steps: object representation, object association, classification, and prediction. Each of the four stages will be discussed below with object association and prediction (tracking) being discussed together as they are tightly coupled, despite being split into two steps.

One major problem encountered with all of the methods reviewed was that they did not consider the presence of static objects in the environment. Therefore, all

detected objects were tracked and unneeded complexity was added to the system. In order to simplify the problem, some approaches filtered out objects that did not fit the expected criteria. In [22] objects that were not on the road were removed from consideration, while in [23] objects that were too large were ignored. Another limitation of the current approaches is that simple shape and motion models are applied and the object's actual dimensions, dynamic properties (maximum velocity, acceleration, etc) and intent cannot be deduced.

**Object Representation**

The first step in any DATMO system is to detect objects in the LADAR scan and represent them in some way. One approach has been to expand upon the occupancy grid concept in order to represent a time varying environment [24] [25] [26]. A time-stamp map is generated in [24], where only cells in the grid corresponding to a laser strike are updated with a time-stamp that indicates the last time the cell was occupied. In [25] each cell is populated with a probability of occupation that is calculated using the new laser data and the previous value.

Clusters, line segments, and rectangular models were all popular representations in the literature and all require a clustering stage. Points are grouped together to represent a single object using one of two general methods: point distance based or Kalman filter based [27], with point distance-based methods being the most popular [28] [29] [30] [31] [32] [33] [34]. The general procedure for the point distance-based clustering algorithm is to compare the distance between successive laser strikes and group points that are within a distance threshold. A number of distance functions are used (such as Euclidean, Mahalanobis, Minkowsky, etc) with the Euclidean distance being the most popular. A novel normalized distance function is introduced in [22] while

[26] uses a K-nearest neighbors approach. The distance threshold which is used for the grouping is usually of two types: fixed or adaptive [35]. The fixed distance threshold approach is simple but does not consider that the distance between scan points are greater for objects that are farther away from the LADAR. Adaptive approaches calculate a maximum distance change between points, based on the distance of a point from the LADAR, which is used for the grouping threshold [35] [27]. Kalman filter based approaches estimate the next point in a cluster, and compare the actual scan point with the estimated point. If the real and estimated points are within the validation gate of the filter the real point is added to the cluster, otherwise a new cluster is started [36] [37] [35] [27]. A novel multiple hypothesis approach to clustering is introduced in [33]. After an initial distance-based clustering, each cluster is considered against a series of possible hypotheses in order to combine clusters that probably belong to the same object.

Once the clusters have been determined, some approaches use the cluster centroid to represent the objects without extracting any features [26] [38] [30] [22] [36] [37]. Additional information is also attached to the object such as velocity or the standard deviation of the cluster [36] [37]. A convex hull is used in [29] to simplify the cluster and compress the data required for storage. It is also used to separate clusters that where erroneously grouped in the initial clustering. The distance from each point to the convex hull is checked and the cluster is split if the distance is greater than a threshold. Other approaches represent the objects as a series of line segments [23] [34] or rectangular bounding boxes [28] [39] [31] [32] [33]. Some papers used the IEPF algorithm to extract line segments [23] [32] [34] while other papers used knowledge

about the expected objects (cars, trucks, etc) to create rules for defining the rectangular bounding boxes [33] [28]. In [39] the rectangular bounding box is aligned to the x-axis in order to account for non-regular shaped objects, where the direction of travel is indeterminate. Other papers attempted to align the bounding box to some assumed object travel direction based on the object geometry. A scheme for merging line segments to account for the legs of people, or walls that appear broken due to occlusion was introduced in [34]. Small line segments were merged if they were separated by a distance of 50cm, while larger line segments where merged if the line segments were collinear and no other line segments existed behind the lines to be merged. One novel innovation presented in [23] improves the object dimension estimates by considering the angular resolution of the LADAR. The closer an object is to the LADAR the greater the number of beams that will strike the object and the smaller the distance between strike points. As such, the detected dimensions will be closer to the real object dimensions. However, the further away the object is the smaller the number of strikes and the greater the distance between strike points. Therefore, the detected width will be less than the real width, which will affect classification (Figure 2-3). In order to alleviate this problem, the object width is estimated using the maximum width that could contain an object (Figure 2-4).

**Object Association and Tracking**

As mentioned, the object association and prediction stages are tightly coupled. The most popular tracking methodology employs a Kalman filter to estimate the new position of the tracked objects [39] [30] [36] [23] [32] [33] [34]. Objects detected in the current scan are then compared to the estimated new position using either a distance measure [26] [39] [38] [30] or a validation region [36]. As the velocity of a new object is

unknown, a large area must be searched to compensate for the error in the first prediction. In order to reduce the number of possible matches after the initial prediction, [32] separates the validation region into preferred areas. In [33] a bounding box overlap and parameter comparison method is used, where measured and estimated objects are associated if the measured object is smaller or equal in size to the estimated object it overlaps, while a network optimization approach is used in [29]. A Bayesian scheme based on a Markov model is used in [40] for tracking. The Markov model developed considers all the possible shapes that can be obtained for a car, bicycle or person when using a LADAR and attempts to match the current scan points to the predefined models.

A novel concept for motion detection is introduced in [38]. The algorithm initially constructs a free space polygon by connecting all the points from the current scan. At each successive time step, the current scan points are compared against the previous free space polygon, and points that fall within the polygon are marked as violation points. The detected violation points are used to extract useful information about the moving object that caused the violations. A heuristic approach is employed to group the violation points into objects that can be tracked over time. When considering grid-based representations, motion detection and tracking was done by comparing grids between time steps. In [24], the previous and current time-stamp maps are compared and cells occupied in both maps contain stationary objects. Cells occupied in the current time stamp map but not in the previous map contain moving objects. A nearest neighbor metric is used to cluster the cells into objects and associate detected objects between time steps.

**Classification**

One interesting addition to some of the DATMO approaches for improving object tracking was the addition of a classification stage. Each detected object was considered against a number of possible classes (cars, pedestrians, etc) and the most probable class was assigned. The dynamics of the assigned class (maximum speed, mobility, etc) was used to aid in the tracking stage. The dimensions of the clusters, line segments, or rectangular bounding boxes were the most commonly used feature in the classification process. However, one of the problems when using a LADAR is that the detected shape of an object varies depending on the position of the object relative to the LADAR [34]. Also, the effects of occlusion affect what is sensed. In order to solve this problem many papers attempted to improve or verify the classification over time by considering previous classifications.

In [31] a priori knowledge about typical road users was used to classify the sensed objects. Typical length and width values were compared against the modeled bounding boxes in order to choose an object class. A verification phase continuously verified the object class using the expected dimensions and the dynamic limitations of the class. The class assignment was changed if another class type became more probable. A similar approach is taken in [34] except that additional features were considered during classification. Each feature detected contributed a weighted “vote” towards a class and the highest scoring class was assigned to the object. One major problem when using the object dimensions during classification is the effect of occlusion. If an object is partially occluded, the object dimensions will be affected and could lead to a false classification. This problem is addressed in [23] by considering object occlusion during the voting process. However, it does not consider object dynamics during the

verification phase like the other papers. A formal probability equation defining the general approach described above is given in [40].

Cameras can also be used for classification [37] [36]. In [36] the camera is the only source of classification, while in [37] LADAR and vision based approaches are performed and fused to provide a more robust classification. In both papers, the LADAR is used to detect objects and generate a Region of Interest (ROI) within the camera image. The generation of the ROI reduces the search area within the image space and therefore reduces computation time, which is a major concern when working with cameras. An AdaBoost classifier is applied to the image to generate a classification which is used in the association stage. In [37] a Gaussian Mixture Model classifier is also applied to the LADAR data. It is important to note that the each classifier is independent of each other and can produce different classifications. The outputs from the two classifiers are then combined using a sum decision rule to make the final classification decision. Experimental results showed a significant increase in hit rate and a decrease in false positives by combining the two classifiers.

### Simultaneous Localization, Mapping and Moving Object Tracking

The idea of considering SLAM in the presence of moving objects is fairly new. A good introduction is given in [41] which considers both the concept of SLAM with generalized objects and SLAM+DATMO and provides justification for the SLAM+DATMO approach. When considering SLAM+DATMO, the biggest question is how to discern between static and dynamic objects in the environment. Moving objects treated as static affect the position estimate while stationary objects marked as dynamic incur unnecessary computational overhead. Temporarily static and slow moving objects introduce another level of complexity as these objects may move while out of sight of

the robot and cause errors in the global SLAM and loop closing approaches. A general approach to the SLAM+DATMO problem is shown in Figure 2-5. All approaches reviewed used a mixed representation approach but most used a grid-based approach as the primary representation strategy. The discussion below will be broken into two sections: heavily grid-based approaches and a hierarchical object representation approach.

Most SLAM+DATMO approaches try to combine classical SLAM and DATMO approaches and suffer from some of the same problems. Grid maps require large amounts of memory and processing power to store and manipulate and the static objects represented have no real world understanding that can aid in the robot's planning stages. The real-life dimensions of moving objects cannot be determined even when the entire object has been sensed and their dynamic properties and future intent cannot be deduced.

**Grid-Based Approaches**

In [42] two grid-maps were generated: one for the static objects and one for the dynamic objects in the environment. In the static object map, each cell contained a probability of occupation, which was calculated using a Bayesian update scheme, while each cell in the dynamic grid stored the number of times a moving object was observed at that cell. Each LADAR strike is categorized as static, dynamic or undecided by comparing the strike against the static and dynamic object maps. If a strike lands in a static cell that is occupied (high occupation probability), the strike is categorized as static; if a strike lands in a static cell that is free space (low occupation probability) or in a dynamic cell that is above a threshold, it is categorized as dynamic; and if a strike lands in a cell where the occupation is unknown, the strike is categorized as undecided.

Strikes categorized as undecided are treated as static until they are proven to be dynamic. A fast scan matching method is introduced to perform data association and estimate the robot's position, which compares the current scan against the built static map and performs a maximum likelihood calculation. Scan points categorized as dynamic are not used to update the map and makes the algorithm resilient to the presence of moving objects, since the probability of cells occupied by moving objects will be low. However, it will be affected by temporarily static objects. The scan points determined to be dynamic are grouped together using a simple distance threshold, and the cluster centroids are used for object tracking. A Kalman filter is used to estimate the new object position and a global nearest neighbor metric associates moving objects between time steps. However, the method does not construct a global map as the main concern is localization in order to facilitate safe navigation, and not mapping, and allows for the use of their fast scan matching algorithm to improve computation time. As no global map is constructed, the loop closing problem is not addressed.

The work done in [43] is very similar to that discussed above and also uses two grid maps. Each cell value in the static object map is the number of times it is occupied by a static object, while each cell value in the dynamic object map is the number of times it is occupied by a moving object. First, the current scan is clustered and compared against the known moving object list. The matching clusters are removed and the remaining clusters are compared against the static object map to generate a position estimate, using the IDC algorithm (an ICP variant) [9]. The updated position estimate is used to find new moving objects from the remaining clusters and the static object map, moving object map, and moving object list are updated. New moving

objects are detected by comparing the new clusters against the static object map and applying two rules. If a previously free area becomes occupied, the cluster in the free space is a moving object (approaching object). If an area that was previously occupied becomes free and an area that was previously occluded is now occupied (leaving object), it is not possible to say that the cluster in the previously occluded area is a moving object (it may be a stationary object that was occluded); however, the area that was previously occupied was occupied by a moving object. Moving object association is done using a similar method to the one used for static object association, which the authors call a matching-based tracking method, since no model is presumed for the moving objects. However, problems exist with this approach which the authors are currently trying to resolve.

Unlike the previous approaches, the work done in [44] only uses a grid-map to represent the static objects in the environment. After the clustering process, known moving objects are removed and the remaining clusters are used to estimate the vehicle position. A correlation approach is used for position estimation by projecting the LADAR strikes into grids at different possible positions and finding the pose that produces the best correlation with the stored static object map. A grid pyramid is employed where a series of grids with different resolutions are stored to improve matching speed. The updated position estimate allows for the separation of the remaining clusters into known static objects and new objects. New objects are first classified as “seed” objects and determined to be moving or static by examining their state histories. Objects that remain “seed” objects for the duration of their observable history (they move out of sensor range) are added to the map as static objects, while moving objects are classified into

one of three categories using a Markov model approach [40]. A novel approach to the SLAM problem developed in this paper is the use of GPS to correct the generated map in a trajectory-oriented closure algorithm. However, the inclusion of the “seed” objects affect the position estimate and lead to loop closure failure.

**Hierarchical Object Representation Approach**

A hierarchical object representation [5] is used in [41] to represent the world. First scan points are clustered using a simple distance criterion and the clusters are associated with the known moving objects using a multi-hypothesis approach. The unmatched clusters are used to detect new moving objects and generate the pose estimate. In order to account for unreliable pose estimation and correspondence, which is present with the ICP algorithm in the presence of sparse data, a sampling and correlation-based range image matching (SCRIM) algorithm is presented. One hundred initial position estimations are generated and correlated to the stored static object maps to find the best correspondence. New moving objects are detected with the application of two separate detectors. The consistency based detector works by comparing the scan points with the surrounding map. A scan point in a free space cell is marked as a moving point, and if the ratio of moving points to total points in a cluster is above a threshold, it is marked as moving. However, slow moving or temporary static objects are difficult to detect, so the moving object map based detector is employed. In this detector a cluster is marked as moving if it is in an area that was previously occupied by a moving object. After the position estimate has been updated and the new moving objects detected, the static object and dynamic object grid maps are updated. Each moving object has a grid associated with it which is updated over time in order to build an object contour. The SCRIM algorithm is used to associate the new scan points with

the object's grid cells while a Kalman filter is used for tracking. The static and moving object grids represent an area that is local to the robot. As the robot moves, new local grids are created and the previous static grids are stored. The stored grid maps are treated as three-degree of freedom features which can be compared using image processing techniques. These stored "features" allows for global SLAM to be performed and the loop closure problem to be solved. However, the stored local grid maps cannot be updated with new position estimates and so overlay consistency cannot be guaranteed.

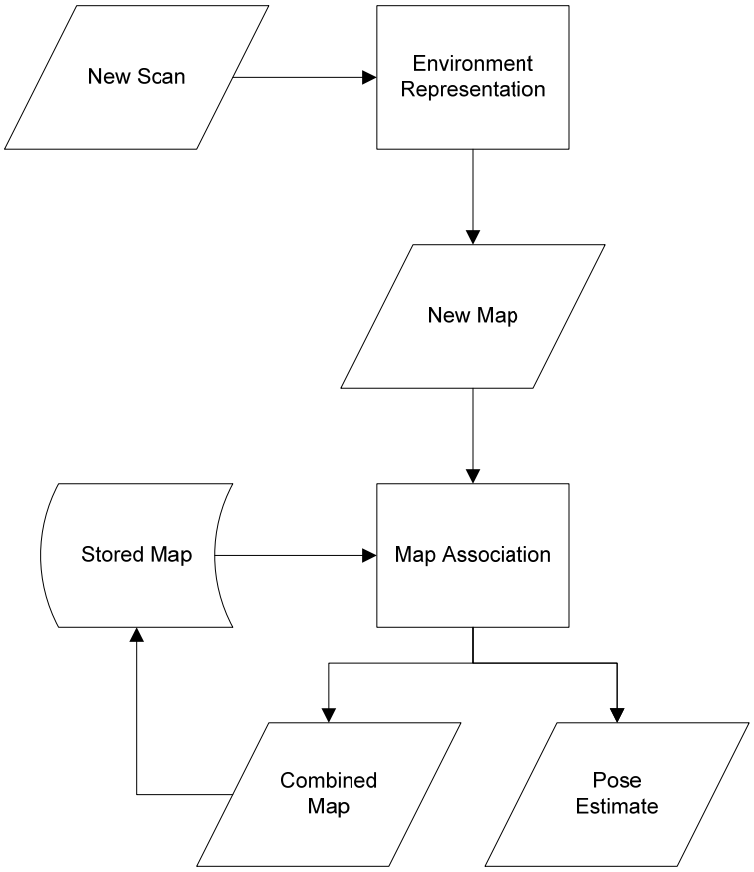



Figure 2-1. Flowchart outlining the general steps in a SLAM system.

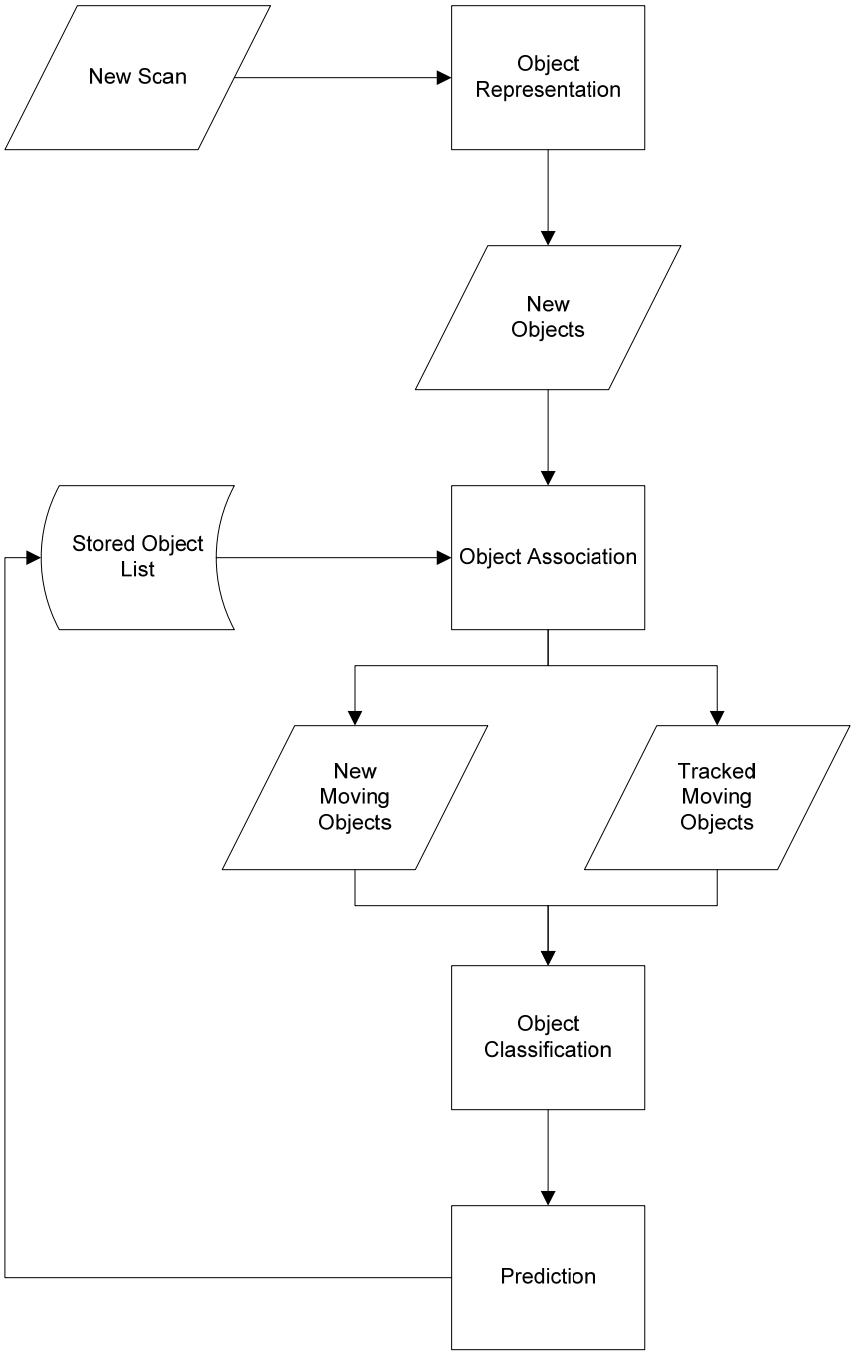



Figure 2-2. Flowchart outlining the general steps in a DATMO system.

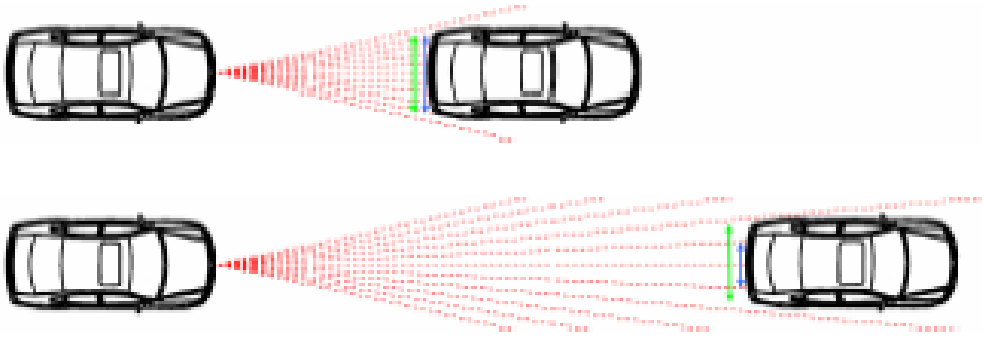


Figure 2-3. Dimension estimate error due to angular resolution of LADAR. The real back segment of the vehicle is shown in green while the detected segment is shown in blue.

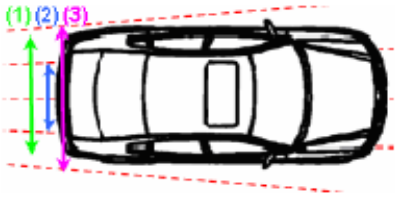



Figure 2-4. Dimension estimate error correction. (1) the real width of the vehicle in green, (2) the detected width of the vehicle in blue, (3) the estimated width in purple.

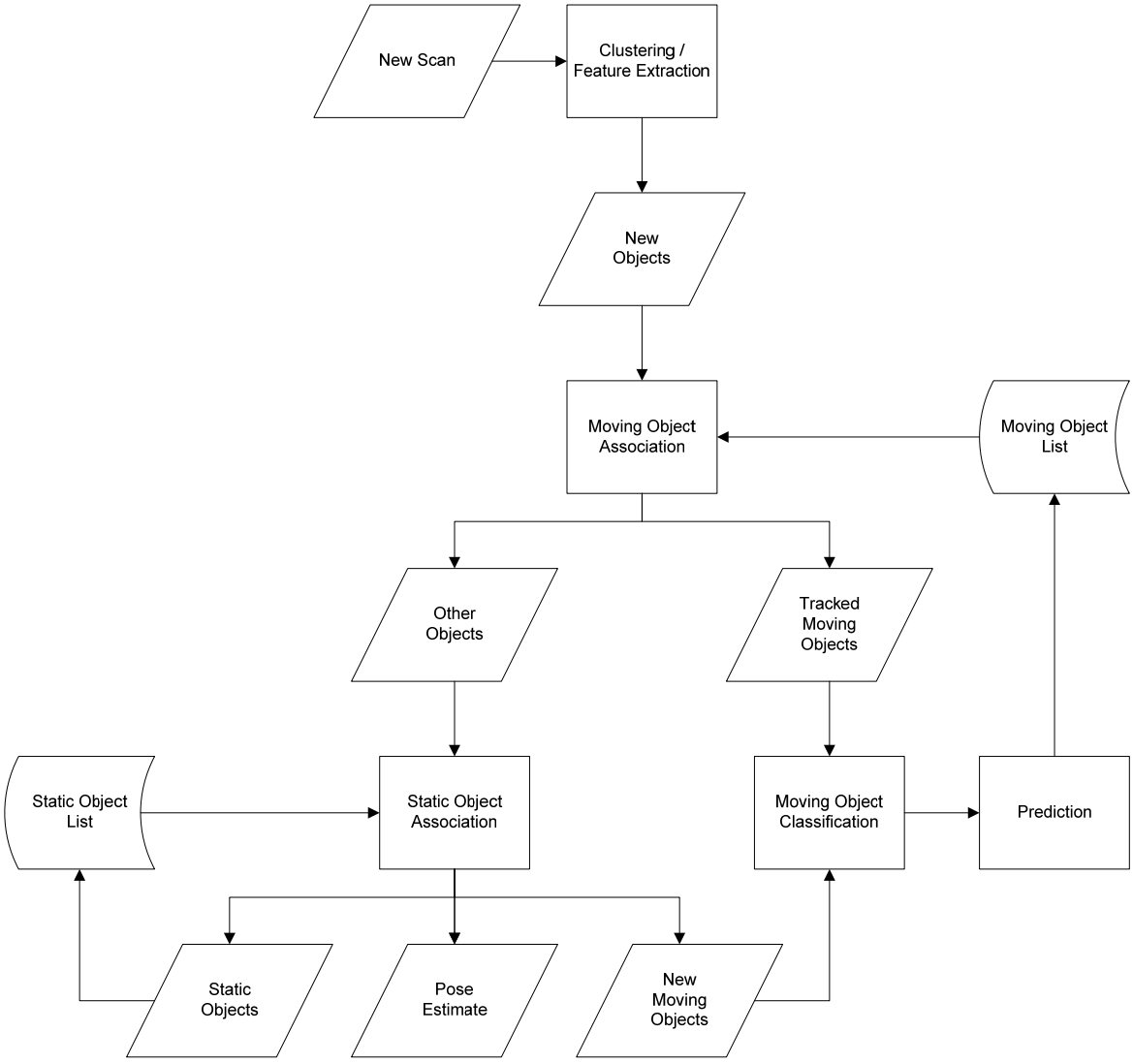



Figure 2-5. Flowchart outlining the general steps in a SLAM+DATMO system.

# CHAPTER 3
# IMPLEMENTED APPROACH

Localization, map generation, and moving object tracking are complex problems and cannot be solved easily. Initially, each problem was approached independently without consideration of the others. The introduction of Simultaneous Localization and Mapping (SLAM) showed that the problems of localization and map generation could be treated together, and that performing both tasks simultaneously generated a better result than when they were tackled separately. Recent work done in combining SLAM with moving object tracking in SLAM+DATMO systems have further shown that by combining all three tasks, better results are obtained. This dissertation introduces a novel approach to the SLAM+DATMO problem which generates an object map instead of the more common point maps, feature maps, or grid maps. It then addresses some of the issues that arise when using a World Model Knowledge Store to store the generated map and introduces a methodology for extending the system for use with multiple LADAR. This chapter serves to introduce the three main elements of the research and provide background for understanding. The first section will present the SLAM+DATMO process and discuss the necessary steps. Next, an introduction into the WMKS will be provided along with a set of requirements to facilitate information exchange. Finally, the methodology for combining multiple LADAR will be discussed. All threshold and parameter values used in the implemented approach are given in Table 3-6.

## Simultaneous Localization, Mapping, and Moving Object Tracking

The topic of SLAM+DATMO is still relatively new and there are still many challenges to be addressed. Most of the work up to this point has involved the use of grid-based approaches for moving object detection and representation. However, grid-

based approaches are generally slower and require more storage space than feature based approaches. Also, previous SLAM and SLAM+DATMO work have treated the mapping aspect simply as a means to perform localization and attempt to ensure map consistency in order to deal with the “loop closing” problem. The topics of map representation and difference detection have not been addressed, although they introduce interesting possibilities with respect to higher-level environmental understanding. The work presented here seeks to address these topics by focusing on three aspects. First, the developed system attempted to represent a singular physical object with a singular representation. For example, a building should be identified as a single polygon or continuous line and not as a collection of points, or a series of unconnected lines. Second, objects are detected and represented completely in the feature-space, that is, grid-based approaches and image processing techniques are not used. In general, grid-based techniques are slower in processing speed and it was believed that it would affect real-time operation, which is a critical requirement when working with a fully autonomous vehicle. Also, grid-based approaches require a lot of storage space which would cause a bottleneck when using the WMKS. Finally, given a saved map, the system attempts to detect differences in the sensed environment and indicate new and missing objects. A flowchart outlining the developed system is shown in Figure 3-1. In this section the different steps of the SLAM+DATMO system will be discussed.

**Object Detection and Representation**

The first challenge in any SLAM+DATMO system is to identify the objects that exist in the environment based on the current LADAR scan. There are generally two steps related to object detection: first, points obtained from the LADAR are clustered

together to identify objects in the environment, and second, features are extracted from the clusters in an attempt to model the object and simplify the object representation. Any reasonable feature can be used but circles, lines and bounding boxes tend to be the most popular. The work presented here models static objects using line segments and moving objects using bounding boxes as they provide reasonable approximations of the objects found in a semi-structured environment, such as a city. As every newly detected object is assumed to be static a description of the method for detecting and representing static objects will be discussed first with a discussion on moving objects given later.

**Clustering**

As mentioned above, the first step in the object detection process was determining which points should be grouped together as part of the same object. Although, there are many different clustering techniques, the work presented used an adaptive distance threshold method [31]. This method considers the distance between two consecutive points to determine cluster membership. If the distance between two points is within a threshold the points are considered to belong to the same cluster, otherwise the points belong to two different clusters. Given two points, $A$ and $B$, taken from the LADAR scan (Figure 3-2), the distance $r_{AB}$, can be calculated by

$$r_{AB} = \sqrt{r_{OA}^2 + r_{OB}^2 - 2r_{OA}r_{OB}cos\alpha}$$(**Error! Use the Home tab to apply Heading 1,00 Chapter Num**

The distance threshold is then calculated by

$$r_{AB} \leq C_0 + C_1 \min\{\mathrm{r_{OA}}, \mathrm{r_{OB}}\}$$(**Error! Use the Home tab to apply Heading 1,00 Chapter Number t**

with

$$C_1 = \sqrt{2(1 - cos\alpha)}$$(**Error! Use the Home tab to apply Heading 1,00 Chapter Number to the te**

The constant $C_0$, allows the algorithm to accommodate for sensor noise and the overlap of pulses at close range while $C_1$ allows for the distance between points to grow as they move further away from the LADAR. One advantage of this technique is that only consecutive points need to be considered. Once a point has been added to a cluster, it is never re-examined during the clustering stage. Another advantage is that the approach is simple and does not require complex calculations. These two factors make the algorithm relatively fast when compared to other methods.

**Feature extraction**

As mentioned, static objects were represented using a series of line segments. Each segment consisted of a start point, an end point, and a variance, where the variance was a fixed value found through experimentation. Tables 3-1 and 3-2 provide a complete list of the fields used. Each line segment point was maintained in the LADAR coordinate frame until processing was complete. The points were stored in either the native polar coordinate system or in the Cartesian coordinate system when in the LADAR frame and using latitude and longitude coordinates when in the global frame.

The Iterative End Point Fit (IEPF) algorithm [23], which is illustrated in Figure 3-3 and described below, was used to extract the line segments from the detected clusters..

1. Initial: Consider a set $S$, consisting of $n$ points.

   $S = \{P_0, P_1, \dots, P_n\}$(**Error! Use the Home tab to apply Heading 1,00 Chapter Number to the**

2. Form a line $L$, between the first and last points in $S$.

   $L = \{P_0, P_n\}$(**Error! Use the Home tab to apply Heading 1,00 Chapter Number to the text t**

3. Calculate the distance from the line $L$ to each point in $S$ to form the set $D$.

   $D = \{d_0, d_1, \dots, d_n\}$(**Error! Use the Home tab to apply Heading 1,00 Chapter Number to the**

4. Find the point with the maximum distance $d_j$ in set $D$.

5. If $d_j > d_{thd}$ split the set $S$ into two sets such that

$$S_1 = \{P_0, P_{j-1}\},\ S_2 = \{P_j, P_n\}$$(**Error! Use the Home tab to apply Heading 1,00 Chapter Numb**

6. Repeat steps 2 to 5 with new sets $S_1$ and $S_2$ until the sets can no longer be separated.

The calculation of the distance $d_k$ between line $L$ and each point $P_k$, in step 3 of the algorithm given above, was calculated by considering the triangle formed between the first point, the last point, and the target point $P_k$, where $d_k$ is the triangle height and the distance between the first and last points is the triangle base, $d_{base}$. As $d_{base}$ can be calculated using Equation 3-1 the distance $d_k$ can be calculated using

$$d_k = \left|\frac{2Area}{d_{base}}\right|$$(**Error! Use the Home tab to apply Heading 1,00 Chapter Number to the text that**

where

$$2Area = |r_0 r_i \sin(\theta_i - \theta_0) + r_i r_n \sin(\theta_n - \theta_i) + r_n r_0 \sin(\theta_0 - \theta_n)|$$(**Error! Use the Home tab to app**

**Object Classification**

The presence of moving objects in the environment causes a number of challenges during the object tracking and localization stages. If they are considered during the matching and resolution stage, it is possible to incorrectly update an object even though it should not be updated. Also, the use of dynamic objects during the localization and mapping stage leads to error. Therefore it is very important to classify the detected objects as either static or dynamic as soon as possible. A free space violation method [38] (Figure 3-4) was used to perform this function. In this method the objects detected in the current scan are compared against a free space polygon generated from the previous LADAR scan.

**Free space polygon generation**

The generation of the free space polygon was performed using a straightforward method that exploited the fact that the points were maintained in polar coordinates. For each point in an object an offset distance, $D_{offset}$ was subtracted from the actual point distance in order to account for sensor noise. For the space between consecutive objects an arc was approximated at the maximum scan distance of the LADAR. Figure 3-5 shows a simplified example of the generation method while Figure 3-6 shows actual system output.

**Moving object detection and representation**

Ideally, any new object that overlaps the free space region should immediately be considered to be a moving object. However, sensor noise and position error can cause static objects to appear to be moving. In order to deal with these inherent errors a probabilistic approach was taken where objects were assigned a confidence that they were static or moving. New objects that were found to overlap the free space polygon where given a positive moving probability while those that did not where assigned a negative probability. During the matching and resolution stage, the overall moving probability of a stored object was updated by adding the moving confidence of the new objects to the confidence of the stored objects. When the confidence of the stored object passed a positive threshold it was classified as a moving object and was treated differently. All objects were treated as static until the moving confidence threshold was met. It was possible for a static object to become a moving object but a moving object could not return to being a static object.

When an object was determined to be moving, its representation was changed from a series of line segments to a bounding box using a list of five points with the first

and last point the same (Table 3-3). The bounding box around an object was determined using a method similar to the one used in [32] where the longest segment from the extracted object was found and treated as the object length. The line segments were then rotated such that the longest segment was aligned to the x-axis. A bounding box was generated by finding the minimum and maximum x and y values and the length and width of the bounding box calculated. If the length or width of the new bounding box was below a minimum size the bounding box was set to the minimum. If the size was less than the previous object size, the previous bounding box was used. The minimum length and width values were chosen from the work presented in [31]. This method ensured a moving object had a minimum size consistent with a car or truck as all moving objects were assumed to be one or the other. It also allowed the bounding box to grow if the detected object was larger than the initial estimate. However, since the box size could not shrink the effect of occlusion could be mitigated. After the bounding box was created all the points were then returned to their original rotation to produce an oriented bounding box. Figure 3-7 gives an example of the bounding box generation process for a moving object.

**Object Tracking**

In the work presented, tracking has two meanings. First, it is the ability to determine if objects that were previously detected still exist in the environment and second, it is the ability to monitor a moving object's motion through the environment. In order to perform tracking, the newly detected objects must be matched to the previously detected objects. An object overlap method was used to perform the matching. If two objects overlapped they were considered to match and therefore were the same object.

**Enclosure generation**

Ideally, static objects that still exist in the environment would always overlap exactly. However, the influence of sensor noise greatly affects the matching process. The points obtained between scans can differ and therefore, a previously detected single object is detected as two objects in the current scan or vice versa. Also, the position of the objects can differ greatly. In order to minimize the influence of sensor noise enclosures were generated around each object. Two enclosure generation methods were considered. First, a simple process similar to the free space polygon generated method which exploited the fact that the points were in polar coordinates was explored. Given a line segment represented by two points such that $L = \{(r_1, \theta_1), (r_2, \theta_2)\}$, the enclosure is represented by four points such that $E = \{(r_1 - v, \theta_1), (r_1 + v, \theta_1), (r_2 + v, \theta_2), (r_2 - v, \theta_2)\}$ where $v$ is the variance of the line from the line extraction stage. However, this method was found to be invalid as it was possible for a line segment to have an enclosure with a zero width if the segment lay along a LADAR scan line. The second method generated the enclosures by using the buffer function in the GEOS library, which is an open source C++ library for modeling and manipulating 2-dimensional linear geometries. This second method was found to be more reliable and robust although slower. It was determined that system robustness was more important that speed and as such the GEOS buffer method was chosen. Figure 3-8 shows examples from both enclosure generation methods.

**Object matching and resolution**

In order to match the stored and new objects every new object was compared to every stored object. Although this method is relatively slow, it produced the most

consistent results when compared to other explored methods. The scenarios that were considered to be possible after matching are listed below and illustrated in Figure 3-9.

- Scenario 1: A new object matches one stored object and the stored object only matches one new object.
- Scenario 2: A new object matches one stored object but the matching stored object matches multiple new objects.
- Scenario 3. A new object matches multiple stored objects but the each of the matching stored objects only match the one new object.
- Scenario 4: A new object matches multiple stored objects but one of the matching stored objects matches multiple new objects.
- Scenario 5: A new object matches multiple stored objects and there exists another new object that matches the same stored objects.
- Scenario 6: A new object matches one old object but that old object overlaps another old object.

Normally, scenarios five and six would not be possible due to the properties of the LADAR. However, the enclosure generation method employed made it was possible for new object enclosures to overlap and therefore these two possibilities could occur. It was found that updating objects based on these scenarios was difficult and undesirable. Therefore, new object enclosures were checked to ensure that they did not overlap and any objects that did were treated as the same object and merged. At first appearance scenarios three and four appear very different to scenarios one and two respectively, however, they can be made the same by merging any old objects than are matched by the same new object during the matching process. In other words, if two stored objects were matched by the same new object, the two stored objects were considered to be the same object. Therefore, scenario three becomes just like scenario one and scenario four becomes like scenario two and the update problem reduces to handling only two possibilities.

First, the update process for scenario one will be considered. When dealing with moving objects the update process is simple. The newly extracted object is converted to a bounding box using the methodology described above. When updating static objects the process is more complex. As mentioned, one goal of the presented research was to generate an object map and therefore, the representation of static objects needed to be updated as more data was obtained. In other words, the object representation needed to be improved if more data was available and previously sensed portions that were occluded or out of the viewable area needed to be preserved. Therefore, the stored static objects could not simply be replaced by the new objects. There are a number of possibilities when considering how the stored and new objects will overlap. The new object could be longer than the stored object and the object length should be increased or the new object could be shorter than the stored object due to occlusion but the overlapping section provides a more accurate representation and should be included. Another problem was determining which line segments should be updated based on the new scan points and which should remain the same. In order to avoid these problems a new cluster was generated based on the stored object and the new object scan points and then the object was regenerated.

One challenge when regenerating the object is maintaining the correct point ordering. Points must be maintained in scan order and therefore, the points cannot be simply grouped together and sorted. This would lead to an ordering that is inconsistent with the LADAR properties and cause the line extraction algorithm to produce bad results. Therefore, the update process consisted of four steps. First, the line segments of the stored object were converted into “pseudo clusters” The clusters were generated

by moving along each line segment at angles consistent with the angular resolution of the LADAR and finding the intersection point between the ray from the LADAR and the line segment (Figure 3-10). Next, the “pseudo-cluster” points from the stored object were associated to the scan points of the matching new object using a closest distance criterion. Third, a new cluster was generated by examining the associated points. Any unassociated points from the stored “pseudo cluster” were added to the new cluster “as is” while associated points were added depending on their position along the object. If the points were in the middle of the object that is they were not the last pseudo point of the stored object, a point corresponding to the average distance between the pseudo points and the new points was added to the cluster. This was done in order to reduce changes caused by sensor noise. If the last pseudo point had numerous new points associated to it, then all the new points would be added to the cluster “as is”. Finally, an object was regenerated from the newly constructed cluster. Figure 3-11 illustrates an example of this process.

When considering a process for updating objects in scenario two it can be seen that the process devised for scenario one can be applied iteratively with the updated object being used in the next iteration. That is, the stored object can be updated using the first new object to generate an improved object representation and then the improved object is updated using the second new object. Using this iterative approach, the update process for scenario two is simplified and the entire update process is reduced to handling the case where a single stored object matches a single new object.

**Missing object detection**

After the matching process, it was possible for some objects to remain unmatched due to sensor noise, platform motion, object motion, object removal or occlusion. It was

considered that there were three categories for an unmatched object: non-visible, occluded, or missing. A non-visible object was one that was outside the viewable range of the sensor and therefore did not intersect with the viewable region polygon. These objects were removed from consideration from the system in order to eliminate unnecessary comparisons during the object matching process. An occluded object was within the viewable region but did not lie inside the free space region and therefore it could not be determined if the object still existed or not. Finally, missing objects were objects that lay within the free space region and were missing from the environment in the current scan. However, the influence of sensor noise caused some objects to be detected sporadically, and since the objects were compared against the free space region from the current scan it was possible for an object to be only missing in the current scan. If an object was removed every time it was not detected objects would constantly be appearing and disappearing. Therefore, a probabilistic approach was taken. An existence confidence was attached to every object to determine if it was missing or not. At each scan, the existence confidence was updated based on if the object was detected, occluded, or missing. If an object's existence confidence dropped below a threshold value it was said to be missing. It is important to note that if the object was occluded its existence confidence could not become negative (or more negative).

**Position Estimation**

When the robot moves from a position $P_{ref}$ to a position $P_{new}$, the difference between $P_{ref}$ and $P_{new}$ is approximately known from the vehicle's positioning system (GPOS). However, the estimate is usually imperfect due to error present in the positioning sensors. The task of the position estimation stage of the SLAM+DATMO

system is to improve the position estimate by aligning the scans taken at $P_{ref}$ and $P_{new}$, referred to as $S_{ref}$ and $S_{new}$ respectively. Assuming the position of the vehicle when scan $S_{new}$ is taken is $P'_{new}$, the position estimation system finds the rotation $\omega$ and translation $T$ for $S_{new}$ such that after the applying the transformation, $S_{new}$ is aligned with $S_{ref}$. However, it is generally impossible to perfectly align the scans due to the presence of sensor noise and occlusion. Sensor noise introduces small deviations which can be characterized using the sensor's inherent error distribution function. On the other hand, occlusion introduces large differences between the scans which cannot be modeled by the error distribution function but can be treated as outlier data and can be ignored.

The estimation method presented here attempts to match the points from the current scan with the modeled static objects from previous scans and was based on the method presented in [9]. It is important to note that only the static objects that were matched during the object matching step were used to generate the position estimate. In general, for each point $P_i$ in $S_{new}$, a simple rule is applied to determine the corresponding point $P'_i$ in $S_{ref}$. A least-squares solution is then computed using the equation:

$E_{dist}(\omega, T) = \sum_{i=1}^{n} |R_\omega P_i + T - P'_i|^2$(**Error! Use the Home tab to apply Heading 1,00 Chapter Nu**

to calculate the relative rotation and translation where

$n$ is the number of corresponding point pairs

$T = (T_x, T_y)$ (**Error! Use the Home tab to apply Heading 1,00 Chapter Number to the text tha**

$R_\omega = \begin{pmatrix} \cos\omega & -sin\ \omega \\ \sin\omega & \cos\omega \end{pmatrix}$(**Error! Use the Home tab to apply Heading 1,00 Chapter Number to th**

The closed form solutions for $T$ and $\omega$ are given by

$$\omega = 2 * \tan^{-1}\left(\frac{Y-\sqrt{X^2+Y^2-Z^2}}{X-Z}\right)$$(**Error! Use the Home tab to apply Heading 1,00 Chapter Number**

$$\hat{x} = \frac{1}{n}\left(-\cos(\omega)\, S_x + \sin(\omega)\, S_y + S_{x'}\right)$$(**Error! Use the Home tab to apply Heading 1,00 Chapter**

$$\hat{y} = \frac{1}{n}\left(\sin(\omega)\, S_x + \cos(\omega)\, S_y + S_{y'}\right)$$(**Error! Use the Home tab to apply Heading 1,00 Chapter N**

where:

$$X = \frac{1}{n}\sum_{i=1}^{n}\left(x_i S_{y'} - y_i S_{x'} - y_i' x_i n + x_i' y_i n\right)$$(**Error! Use the Home tab to apply Heading 1,00 Cha**

$$Y = \frac{1}{n}\sum_{i=1}^{n}\left(-x_i S_{x'} - y_i S_{y'} + x_i' x_i n + y_i' y_i n\right)$$(**Error! Use the Home tab to apply Heading 1,00 Ch**

$$Z = \frac{1}{n}\sum_{i=1}^{n}\left(-x_i S_y + y_i S_x\right)$$(**Error! Use the Home tab to apply Heading 1,00 Chapter Number to**

and

$$\mathrm{S_y} = \sum_{i=1}^{n} \mathrm{y}_i$$(**Error! Use the Home tab to apply Heading 1,00 Chapter Number to the text that**

$$S_{y'} = \sum_{i=1}^{n} y_i'$$(**Error! Use the Home tab to apply Heading 1,00 Chapter Number to the text tha**

$$S_x = \sum_{i=1}^{n} x_i$$(**Error! Use the Home tab to apply Heading 1,00 Chapter Number to the text that**

$$S_{x'} = \sum_{i=1}^{n} x_i'$$(**Error! Use the Home tab to apply Heading 1,00 Chapter Number to the text tha**

The new rotation and translation is then applied to the current position estimate to reduce the position error between the two scans and the process is repeated until the solution converges.

The method described above has been used extensively for position estimation in point map approaches where every point is maintained. However, the presented research generates an object map not a point map and therefore, the method cannot be directly applied. Fortunately, the method used for object resolution already converts the stored line segments into points and associates the stored and new points. Therefore, once the points have been associated, the present SLAM method can be applied directly. The convergence criterion used was simply a set number of iterations of the

algorithm. Multiple iteration thresholds were considered but through experimentation it was found that 20 iterations gave the best results when considering accuracy versus speed. It was assumed that the error of the initial position estimate produced by GPOS was small and would not greatly affect the matching process. Therefore, object matching was only performed once while the stored and new object points were re-associated during every iteration.

## World Model Knowledge Store

Although, the use of knowledge stores for centralized storage is not new to the field of robotics, none of the work reviewed attempted to use an external central repository. In fact, most of the systems did not address the issue of data sharing at all and were architected to be completely self contained. The research presented here attempts to address this shortcoming by providing a mechanism for disseminating information about detected objects to other components of the autonomous vehicle. The inclusion of such a mechanism facilitates modularity and component reuse among autonomous systems; two important aspects within the field of robotics.

The exact implementation details of the WMKS are outside of the scope of the research presented here but was based on the work done in [46]. The WMKS backend was implemented using a PostgreSQL database extended for geo-spatial support using PostGIS. Data exchange was facilitated through the use of messaging based on a draft version of the AS-4 World Modeling Knowledge Store standard with modifications as needed.

### Challenges

There are two approaches that can be taken when communicating with the external WMKS: a synchronous approach and an asynchronous approach. In the

synchronous approach when a command is sent to the WMKS the client waits until it is completed before continuing operation. In the asynchronous approach the client sends the command and continues processing without waiting. The client may or may not need confirmation that the command actually completed. In general, the synchronous approach is easier to understand and implement. However, when dealing with real-time systems transmission latency becomes a major concern. If latency is high, the client may wait a long time before the command is completed and real-time operation is affected. The effect of transmission latency is decreased in an asynchronous approach but introduces problems with data synchronization as there will be some period of time when the WMKS is out of date. In the presented research real-time operation was deemed critical. Delays in processing could result in one of two possibilities: LADAR data is skipped or lost during the wait period, or the system processes old LADAR data that no longer represents the current environment. Neither scenario was desirable and could lead to unsafe behavior by the autonomous vehicle. Therefore, an asynchronous approach to WMKS communication was taken. In order to combat the issues of data synchronization, a local storage cache was used to store objects of immediate interest. All objects within range of the LADAR were stored in the local cache and the cache was synchronized periodically with the WMKS. In addition a confirmation scheme was used to improve robustness. In this scheme, every command elicits a response from the WMKS and allows the client to recover from synchronization errors.

Transmission loss was another major concern when dealing with WMKS communication. If a command is sent to the WMKS and the confirmation is not received, it is impossible to determine if the failure was as a result of transmission loss

or by some problem in the WMKS. Improper handling of the failure could lead to discrepancies between the local cache and the WMKS. Say a create command is sent to the WMKS but is not received due to transmission loss. If the client resends the create command, the system recovers. However, if the client assumes the command was successful, the cache and the WMKS are out sync. Conversely, if the command successful completed but the confirmation was never received, the WMKS and the cache are in sync if the client assumes transmission loss. However, if the client resends the create command, the WMKS contains two copies of the objects and becomes out of sync with the local cache. The issues surrounding transmission loss were considered to be outside the scope of the research presented here. Therefore, in order to deal with the described challenges, guaranteed data delivery was assumed. If a command sent to the WMKS did not execute it was assumed that it was caused by WMKS failure and not by transmission loss.

**Messaging**

In order to add, modify, or delete objects in the WMKS a mechanism for transmitting data between the computing resources was required. It was determined that a messaging scheme would be used to send commands to and receive responses from the WMKS. There were three main commands the WMKS needed to handle: adding an object, modifying an object, and querying an object. It was determined that objects would not actually be deleted from the WMKS but instead updated to a special status to indicate deletion. This functionality provided an easy method for checking the correctness of the SLAM+DATMO system. If objects were deleted it would be impossible to determine if the system erroneously deleted a detected object or if the object was never detected. The six messages needed to handle the required

functionality are described below. However, before the messages can be introduced an overview of the object representation within the messages is needed.

**Message object description**

The objects stored within a WMKS can vary widely depending on the requirements of the implemented system and introduces an interesting challenge when considering the messaging between a client and the WMKS. One approach would be to implement custom messages for every possible object type the WMKS supports. However, this approach leads to major changes when support for a new object or modifications to existing objects are required. A second approach would be to have a limited number of general messages that can be used for any object type. The object would be described within the message with a format that can be generated, parsed, and understood by the client and the WMKS. The second approach is used within the implemented WMKS messaging scheme. An object is represented by a series of properties or attributes, where each attribute is described using three fields.

- **Attribute ID.** An enumeration that uniquely identifies what the attribute represents. E.g. height, weight, centroid, color, outline.
- **Attribute data type.** An enumeration that uniquely identifies the type of the data used to represent the attribute. E.g. double, integer, point, polygon, list.
- **Attribute data.** The actual value of the attribute, which can also be a list of attributes. E.g. A polygon is a list of points.

Enumerations are used for the Attribute ID and the Attribute Data Type as a means of reducing the message size. String descriptions could be used instead of enumerations but would lead to larger messages and are more difficult to parse.

**Create Knowledge Store Objects message**

The Create Knowledge Store Objects message is used to add new objects to the WMKS and consists of three mandatory fields.

- **Message properties.** The message properties field is used to request a certain behavior from the WMKS when it receives the Create message. There are five possible behaviors based on the value of the field.
    - No response. The WMKS will do nothing after the objects have been created.
    - Confirm create. The WMKS will send a Report Knowledge Store Objects Creation message after the objects are created.
    - Confirm object count. The WMKS will indicate the number objects that were successfully created in the Report message.
    - Confirm WMKS object IDs. The WMKS will provide the list of the WMKS Object IDs for all the created objects.
    - Confirm WMKS objects. The WMKS will provide the list of objects that were created including their WMKS Objects IDs.
- **Request ID.** A client set enumeration that can be used to correlate a Create message with its corresponding Report message.
- **WMKS object list.** The list of objects that need to be created.

**Report Knowledge Store Objects Creation message**

The Report Knowledge Store Objects Creation message is used in response to the Create Knowledge Store Objects message to indicate if the requested object additions were successful. The message consists of two mandatory fields and three optional fields.

- **Presence vector.** This field indicates which of the optional fields are provided in the message.
- **Request ID.** The Request ID of the original Create message.
- **WMKS object count.** An optional field that indicates the number of the objects that were successfully created.

- **WMKS object ID list.** An optional field that provides the WMKS Object IDs of all the created objects in the order they were provided in the Create message. Objects that could not be created are indicated with a WMKS Object ID number of zero.

- **WMKS object list.** An optional field that lists all the WMKS Objects that were successfully created. Objects that were not created are not listed.

**Modify Knowledge Store Objects message**

The Modify Knowledge Store Objects message is used to update existing objects in the WMKS and consists of four mandatory fields.

- **Message properties.** The message properties field is used to request a certain behavior from the WMKS when it receives the Modify message. There are two possible behaviors based on the value of the field.

    - No response. The WMKS will do nothing after the objects have been modified.

    - Confirm modify. The WMKS will send a Report Knowledge Store Objects Modify message after the objects have been modified.

- **Request ID.** A client set enumeration that can be used to correlate a Modify message with its corresponding Report message.

- **Query filter.** A description of the objects that need to be modified within the WMKS. The filter is constructed using the Attribute ID, Attribute Data Type, and Attribute Value fields but also allows for complex queries involving AND, OR, and NOT operations.

- **Attribute list.** A list of attributes that need to be updated for all objects that matched the query filter.

**Report Knowledge Store Objects Modify message**

The Report Knowledge Store Objects Modify message is used in response to the Modify Knowledge Store Objects message to indicate if the requested object updates were successful. The message consists of three mandatory fields.

- **Modify success.** A boolean value that indicates if the query filter in the Modify message was valid.

- **Request ID.** The Request ID of the original Modify message.

- **WMKS object count.** The number of objects that were modified.

**Query Knowledge Store Objects message**

The Query Knowledge Store Objects message is used to search for existing objects in the WMKS and consists of four mandatory fields and one optional field.

- **Presence vector.** This field indicates which of the optional fields are provided in the message.
- **Message properties.** The message properties field is used to request a certain behavior from the WMKS when it receives the Query message. There are two possible behaviors based on the value of the field.
    - No additional processing. The WMKS will return only the objects that match the specified query.
    - Return dependent objects. The WMKS will return the dependents for the objects that match the specified query. A dependent is a WMKS object that is attached to another WMKS object. For example, if the WMKS stores Wheel objects and Car objects separately, then the Wheel object could be a dependent of the Car object, since a Car has four wheels.
- **Request ID.** A client set enumeration that can be used to correlate a Query message with its corresponding Report message.
- **Query filter.** A description of the objects that need to be reported from the WMKS. The filter is constructed using the Attribute ID, Attribute Data Type, and Attribute Value fields but also allows for complex queries involving AND, OR, and NOT operations.
- **Return filter.** An optional field that lists the object attributes that should be returned in the Report message. The use of the Return Filter allows a client to only deal with attributes important to the client and helps reduce message size.

**Report Knowledge Store Objects message**

The Report Knowledge Store Objects message is used in response to the Query Knowledge Store Objects message to list the objects that matched the query. The message consists of two mandatory fields.

- **Request ID.** The Request ID of the original Query message.

- **WMKS object list.** The list of WMKS Objects that match the objects within the Query message.

**Updated SLAM+DATMO Object Representation**

When using the WMKS, additional fields must be added to the previously discussed SLAM+DATMO object representations. These fields fall into one of two categories: fields that the SLAM+DATMO system uses to monitor the status of the object in the WMKS, and fields that are required by the WMKS. Monitoring the object status in the WMKS is important for two purposes. First, it prevents duplicate objects. If the SLAM+DATMO system does not know that an object has been sent to storage then it may attempt to store the object again. The WMKS is a simple storage mechanism and does not check for duplication or perform any form of object resolution. Therefore, it is the SLAM+DATMO system's responsibility to ensure duplication does not occur, especially since it will adversely affect system performance. The addition of a storage status field performs this purpose. When an object is sent to the WMKS, its status is changed to SENT_TO_STORAGE which tells the system that it should not send it again. When the object is successfully stored in the WMKS, its status is changed to STORED and the system is now able to send modification requests.

The second reason for monitoring the object's status is to minimize traffic between the WMKS and the SLAM+DATMO system. The SLAM+DATMO system can potentially run at 10Hz when using a single LADAR and objects may be updated at that rate. If the objects in the WMKS were updated every cycle there would be a large amount of traffic between the WMKS and the SLAM+DATMO system which would prevent requests from other components from being processed. In order to minimize data traffic three fields were introduced: update count, update time, and confirm time. The update count field

was used to count the number of times the object had been updated by the SLAM+DATMO system, while the update time field tracked the last time a request to update the object was sent to the WMKS. The confirm time field was used to track the time the last WMKS update was successfully performed. There were two criteria for updating an object in the WMKS. Updates were requested immediately when certain special fields were changed, such as a change of the identification status to DUPLICATE. Updates to other fields did not immediately generate a request but were sent to the WMKS if the last update request was successfully completed, the object had been modified numerous times, and a sufficient period of time had passed since the last confirmed update. The additional fields and threshold values used are given in Tables 3-4 and 3-6 respectively. These additional monitoring fields were not stored with the object in the WMKS and were only used internally to the SLAM+DATMO system.

In addition, the WMKS adds two fields to every stored object which are listed in Table 3-5. The WMKS ID field is a unique ID for all objects within the WMKS while the WMKS Object Type is a field that is used to determine what type of object is being described. The WMKS object type field is important for message parsing and for converting the WMKS objects to the SLAM+DATMO representation while the WMKS ID field is never used as the Object ID field is more useful.

### Laser Range Finder Fusion

A popular approach to LADAR fusion in the past has been through the use of grid-based methods where a grid is generated and updated by both LADAR independently. Although reasonable, grid-based approaches were deemed not appropriate within the context of the research presented here, which seeks to avoid complex image processing methods. Recently, some work using feature-based fusion approaches have

been explored but they have been limited to static environments using a stationary platform. The work presented here extends the shared object fusion method outlined in [45] to include the use of a moving platform in the presence of a dynamic environment. Data from each LADAR is processed separately but synchronously. The local object cache was shared between the two LADAR but was only updated by one LADAR at a time and data from one LADAR was only processed if no processing was being performed on the other. Objects generated in the current scan were only compared to the free space polygon generated by the last scan from the same LADAR. Therefore, it was important that both LADAR be processed before any one LADAR is processed again. The synchronous nature of the fusion method alleviated the challenges of data coherence and conflict resolution which occur with asynchronous approaches. In order to further simplify the problem, it was assumed that both LADAR scan along the same plane which is at a zero degree inclination to the vehicle's x-y plane. As each LADAR is processed independently it was not assumed that the scan data was taken at the vehicle's current position. Instead, a history of the vehicle's position was maintained and the LADAR scan time was used to determine the global position of the LADAR when the scan was taken.

s

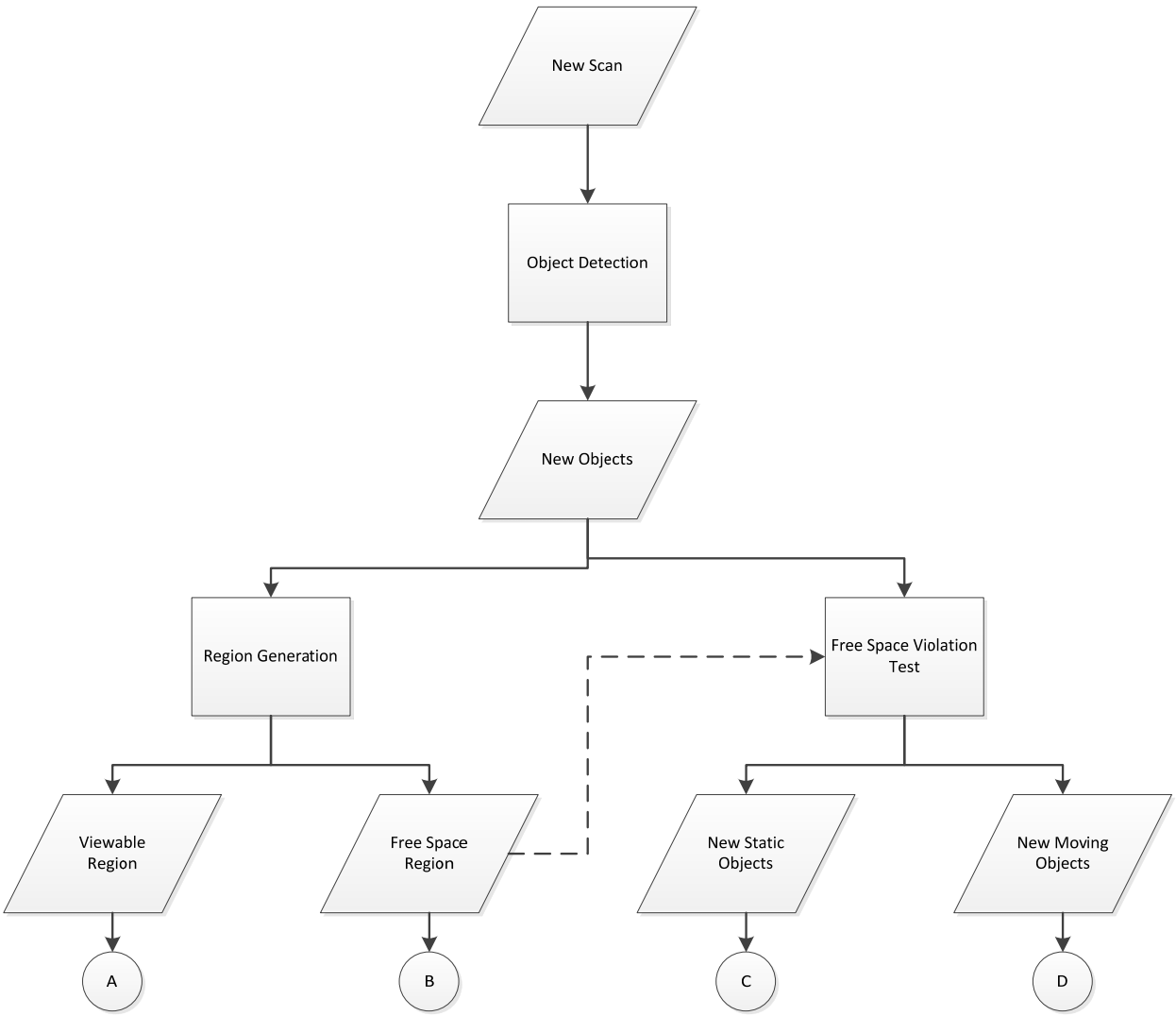



Figure 3-1. Flowchart outlining the presented approach.

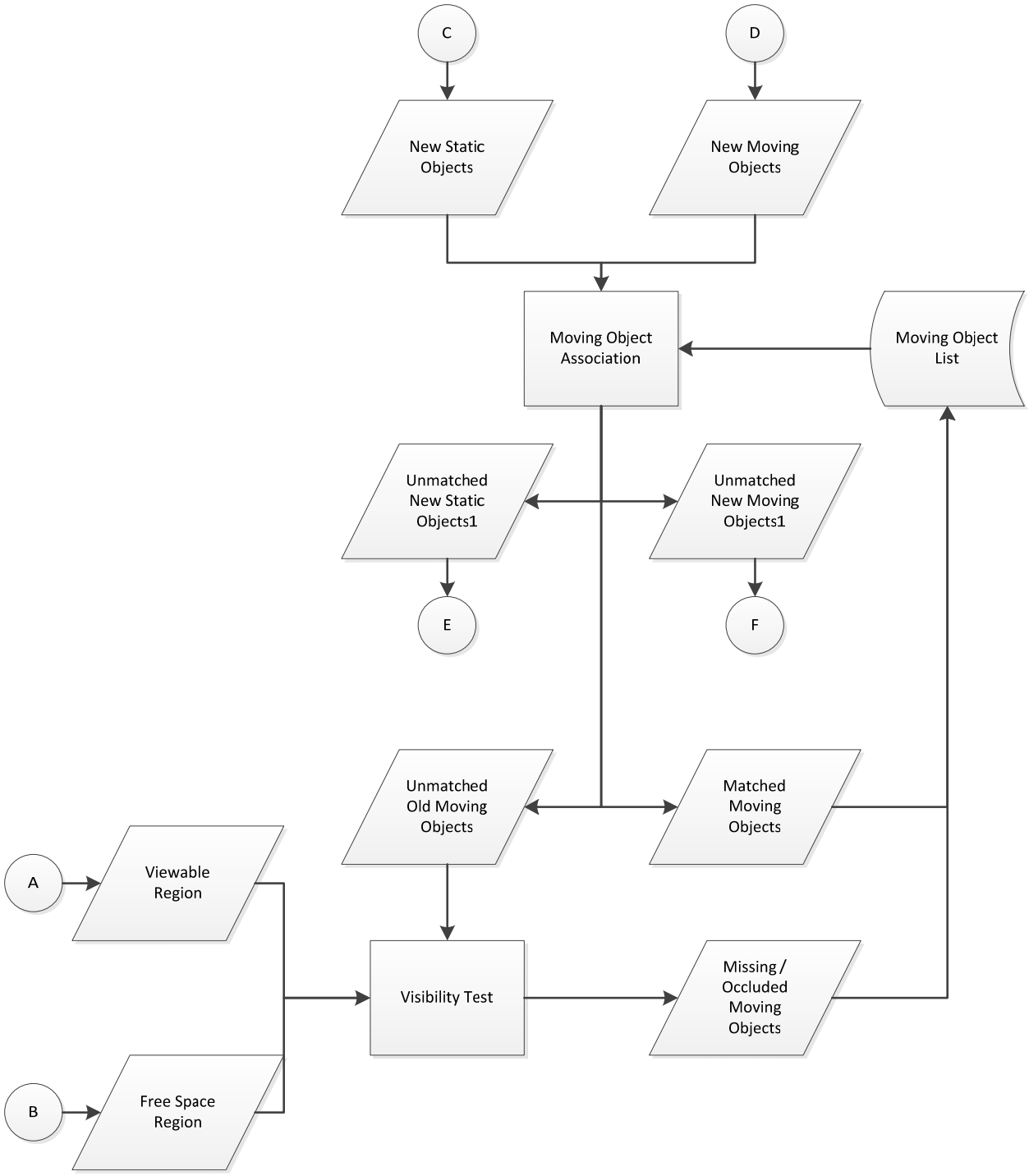



Figure 3-1. Continued.

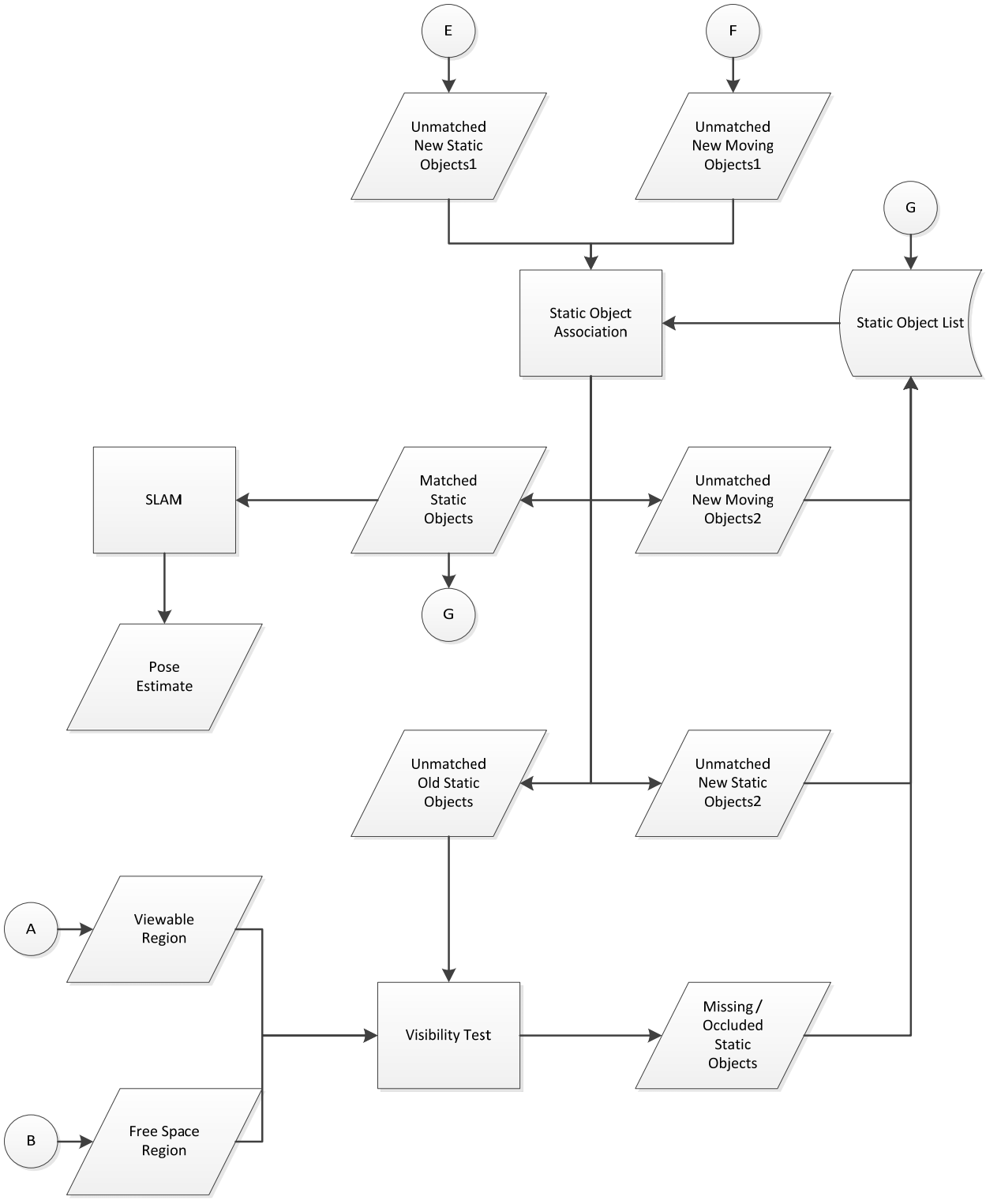



Figure 3-1. Continued.

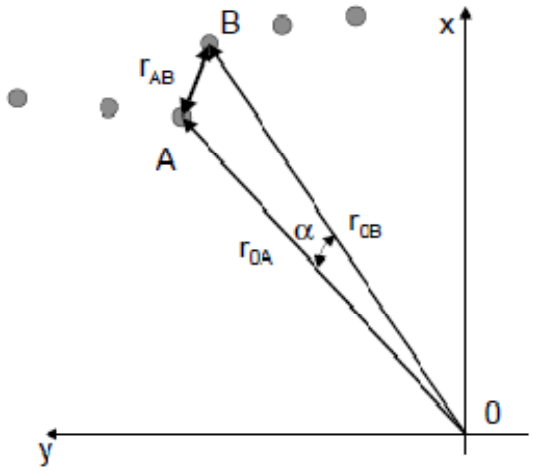



Figure 3-2. Example of the clustering process. If the distance $r_{AB}$ between points A and B are within a threshold distance, the points are considered part of the same cluster.

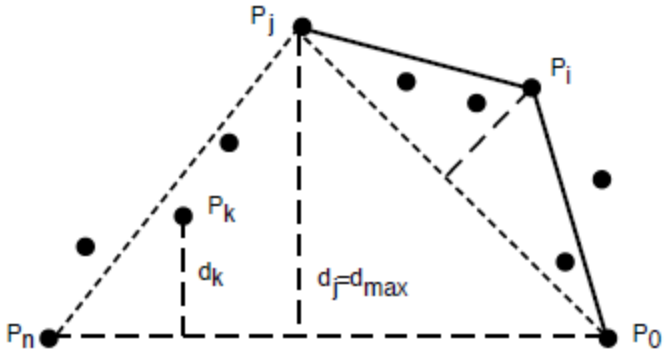



Figure 3-3. The Iterative End Point Fit (IEPF) Algorithm. The algorithm searches for the point $P_j$ with the greatest distance from the line through $P_0$ and $P_n$. If the distance is greater than the threshold T, the line $P_0P_n$ is broken into two lines $P_0P_j$ and $P_j$Pn and the process repeated for the two new lines.

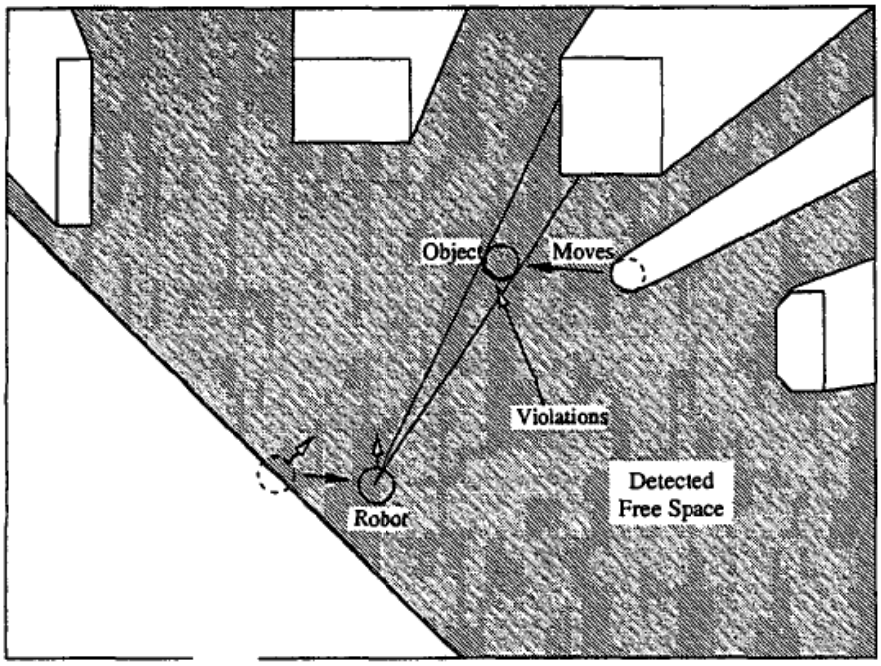



Figure 3-4. Example of the moving object detection method. The grey area represents the free space region detected using the LADAR while the white area represents the occluded region. When the object moves it overlaps the free space region and is identified as a moving object.

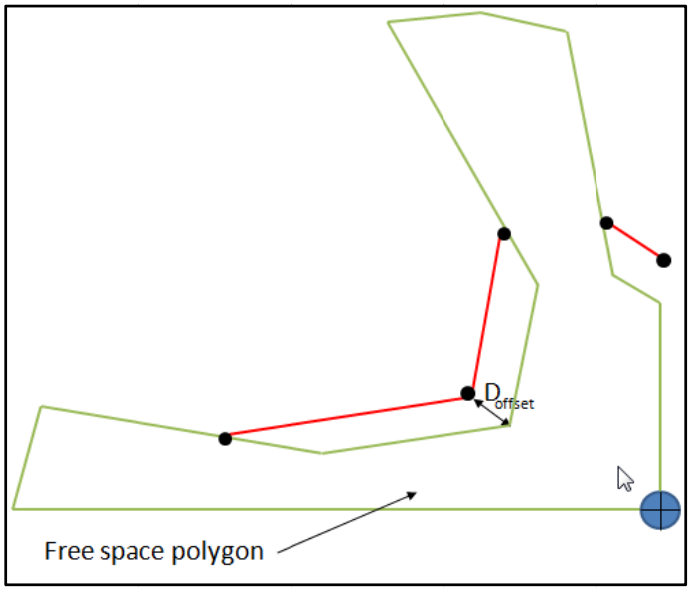



Figure 3-5. An example of the free space polygon generation method. The blue circle represents the LADAR while the red lines represent the detected objects. The green lines outline the generated free space polygon.

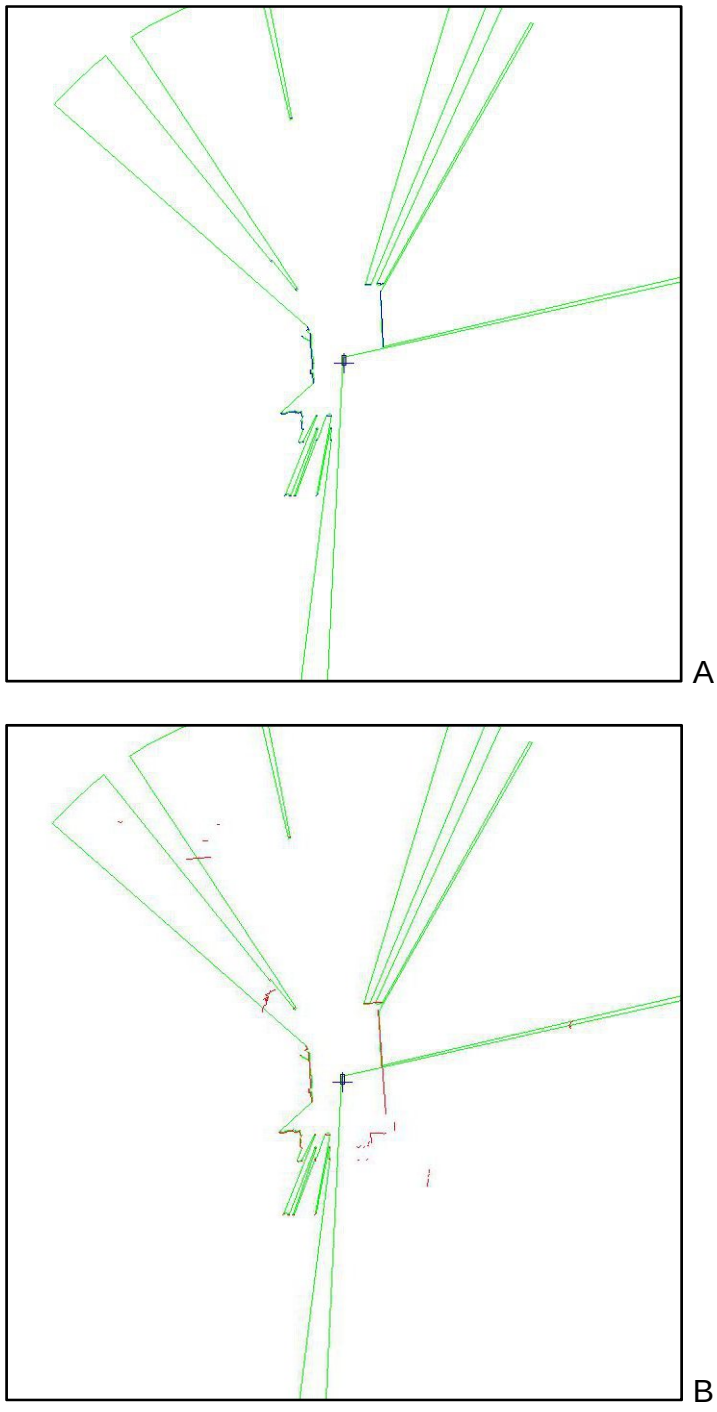



Figure 3-6. Free space region generated around the vehicle. A) The extracted objects are used to generate the free space region. B) The free space region overlaps objects detected in previous scans. The objects overlap the free space polygon due to sensor noise, or removal from the environment.

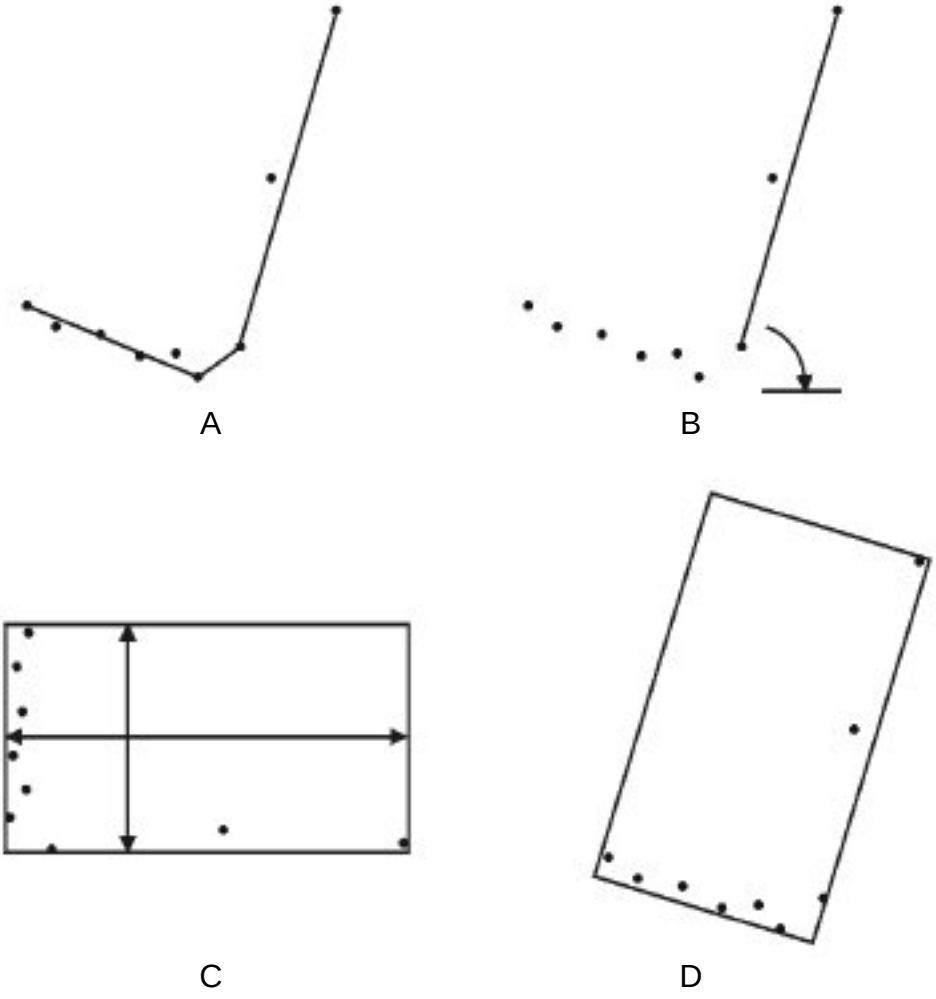



Figure 3-7. Generation of the oriented bounding box used for moving object representation.

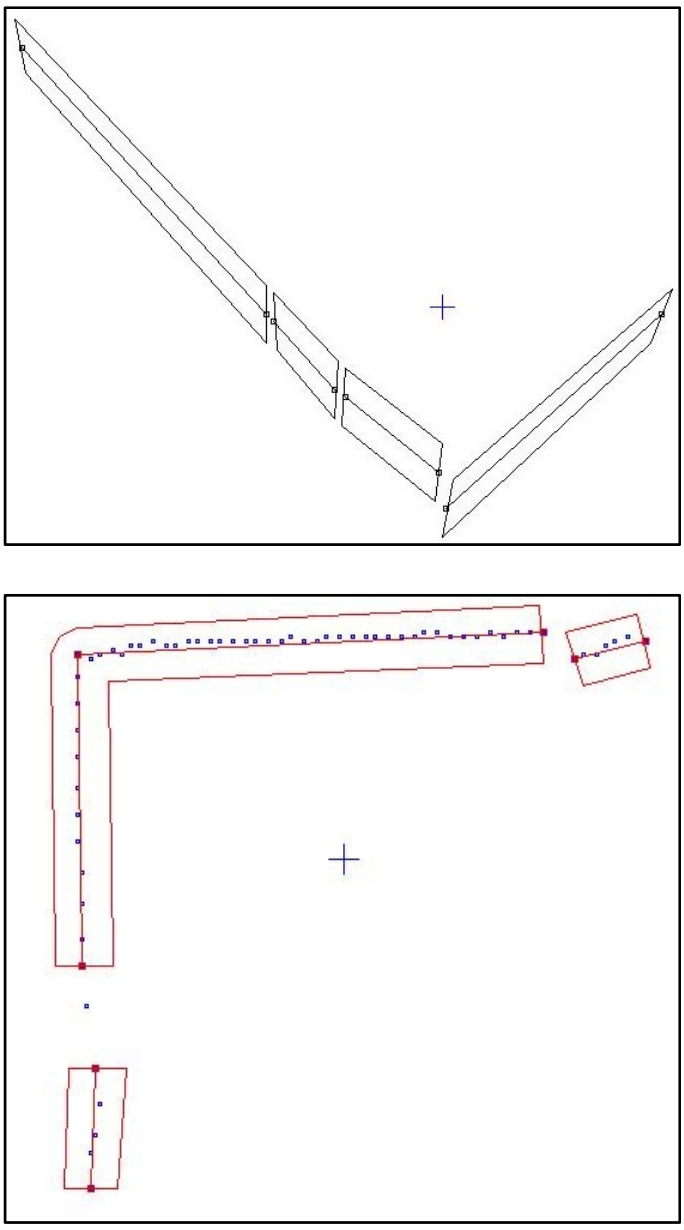



Figure 3-8. Example of the enclosures generated around the line segments. A) Enclosures generated in LADAR's polar coordinate system. B) Enclosures generated using the GEOS library's buffer function.

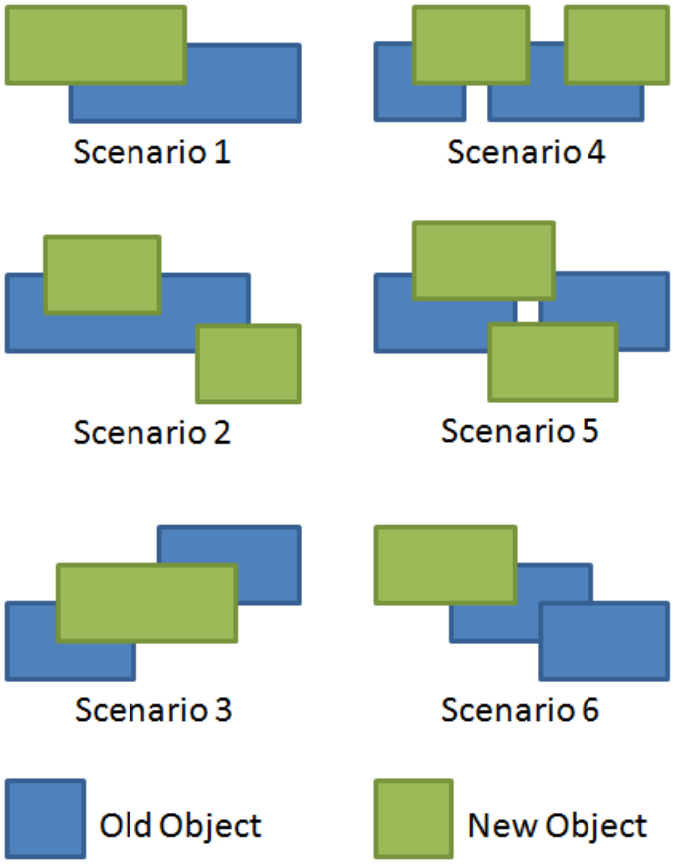



Figure 3-9. Possible scenarios that can occur after object matching.

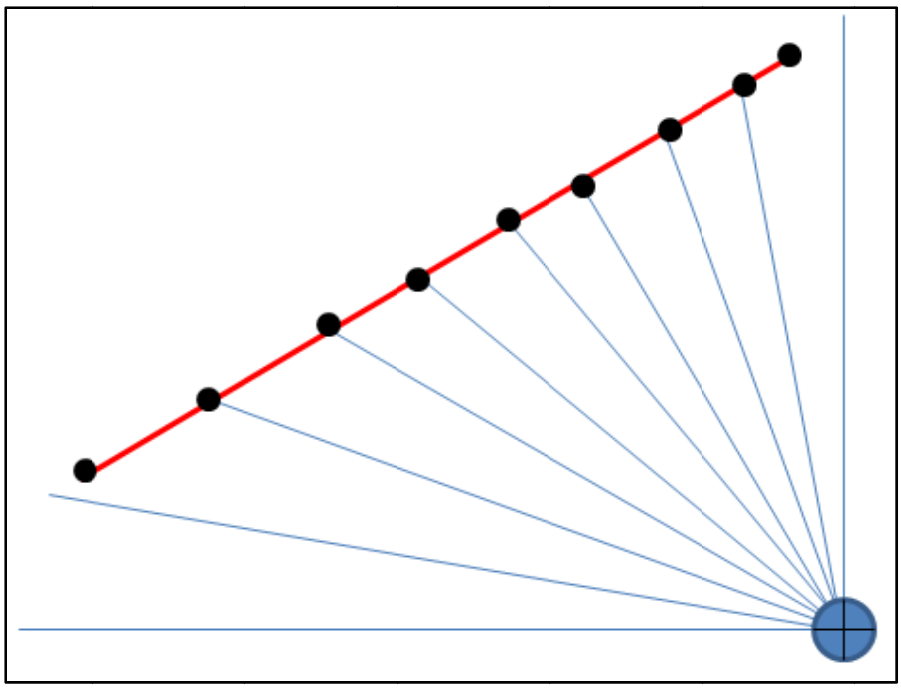


Figure 3-10. “Pseudo-cluster” points (black) are generated from the line segments of the stored objects.

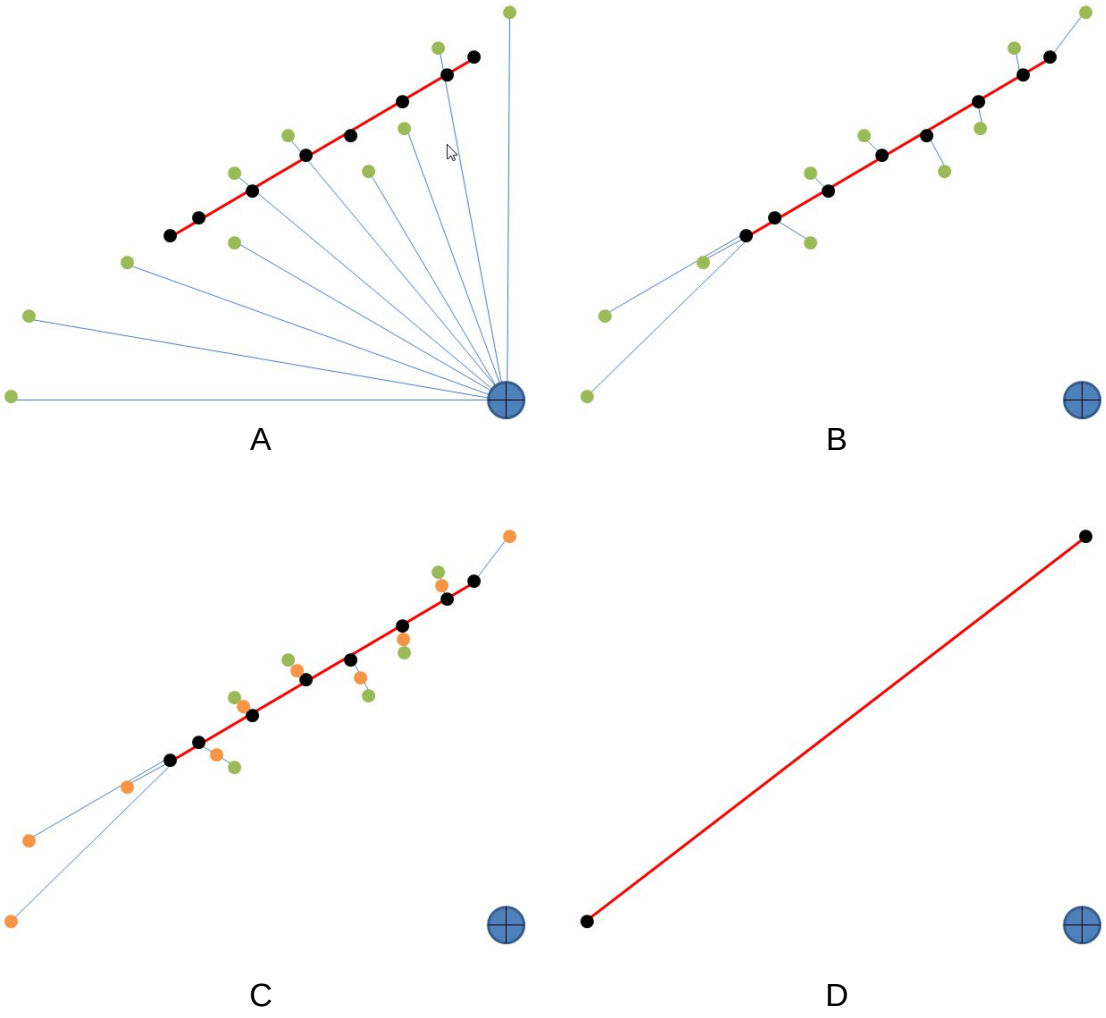



Figure 3-11. Object resolution example. A) “Pseudo-cluster” points (black) are generated along the stored objects (red). B) The current scan points (green) are associated with the “pseudo cluster” points based on the closest distance. C) Points are added to a new cluster (orange) based on their position along the object. D) A new object is generated from the new cluster which produces a better representation of the object.

Table **Error! Use the Home tab to apply Heading 1,00 Chapter Number to the text that you want to appear here.**-1. Fields used to represent a static object

| Field Name | Field Data Type | Description |
|---|---|---|
| Object ID | Unsigned Integer | An identifier that uniquely enumerates the objects detected by the system. |
| Geometry | List of Line Segments | A list of line segments that represents the shape of the object. |
| Centroid | Point | A point that represents the centroid if the object geometry. |
| Existence Confidence | Integer | A confidence value that indicates if the object still exists or not. |
| Moving Confidence | Integer | A confidence value that indicates if the object is moving or not. |
| Identification Status | Byte | The moving object's identification status. The status can be one of four values: UNKNOWN, KNOWN, MISSING, and DUPLICATE. |

Table **Error! Use the Home tab to apply Heading 1,00 Chapter Number to the text that you want to appear here.**-2. Fields used to represent a line segment

| Field Name | Field Data Type | Description |
|---|---|---|
| Start Point | Point | The first point of the line segment |
| End Point | Point | The last point of the line segment |
| Variance | Double | The variance of the points used to generate the line |

Table **Error! Use the Home tab to apply Heading 1,00 Chapter Number to the text that you want to appear here.**-3. Fields used to represent a moving object

| Field Name | Field Data Type | Description |
|---|---|---|
| Object ID | Unsigned Integer | An identifier that uniquely enumerates the moving objects detected by the system. |
| Geometry | List of Line Segments | A list of line segments that represents the shape of the object. |
| Length | Double | The length of the moving object. |
| Width | Double | The width of the moving object. |
| Geometry | List of Points | A list of points that represents the bounding box of the moving object. A list must contain five points where the first and last points are the same. |
| Centroid | Point | A point that represents the centroid of the moving object geometry. |
| Existence Confidence | Integer | A confidence value that indicates if the object still exists or not. |
| Moving Confidence | Integer | A confidence value that indicates if the object is moving or not. |
| Identification Status | Byte | The moving object's identification status. The |

| | | |
|---|---|---|
| | | status can be one of four values: UNKNOWN, KNOWN, MISSING, and DUPLICATE. |

Table **Error! Use the Home tab to apply Heading 1,00 Chapter Number to the text that you want to appear here.**-4. Additional fields needed when using the WMKS

| Field Name | Field Data Type | Description |
|---|---|---|
| Storage Status | Byte | The object's storage status in the WMKS. The status can be one of three values: Not Stored, Sent to Storage, and Stored. |
| Update Count | Integer | The number of times the object has been modified. |
| Update Time | Double | The last time the object was updated in the WMKS. |
| Confirm Time | Double | The time the last WMKS update was confirmed. |

Table **Error! Use the Home tab to apply Heading 1,00 Chapter Number to the text that you want to appear here.**-5. Fields added to all WMKS Objects

| Field Name | Field Data Type | Description |
|---|---|---|
| WMKS ID | Unsigned Integer | An identifier that uniquely enumerates the objects within the WMKS. |
| WMKS Object Type | Byte | An enumeration that indicates the type of the object that is being represented. |

Table **Error! Use the Home tab to apply Heading 1,00 Chapter Number to the text that you want to appear here.**-6. Threshold values and parameters used in the SLAM+DATMO system

| Parameter Name | Value | Description |
|---|---|---|
| Minimum Scan Distance | 0.0 m | The minimum distance a point can be from the LADAR for it to be considered a valid point. |
| Maximum Scan Distance | 200.0 m | The maximum distance a point can be from the LADAR for it to be considered a valid point. |
| Minimum Cluster Distance, $C_0$ | 0.25 m | The minimum distance between two scan points for them to be considered part od the same cluster. |
| Line Break Distance, $d_{thd}$ | 0.25 m | The maximum distance a point can be from an extracted line segment before the line segment will be sub-divided. |
| Line Segment Variance | 0.40 m | The variance assigned to every line segment of an object. This value is used when generating the object enclosures. |
| Laser Angle Error | 0.16 degrees | The error associated with the angular accuracy of the LADAR. |

| | | |
|---|---|---|
| Region Distance Offset, $D_{offset}$ | 0.60 m | The buffer distance between the free space polygon and the extracted objects. |
| Minimum Moving Object Length | 4.0 m | The minimum length of a moving object. |
| Minimum Moving Object Width | 2.0 m | The minimum width of a moving object. |

Table **Error! Use the Home tab to apply Heading 1,00 Chapter Number to the text that you want to appear here.**-6. Continued.

| Parameter Name | Value | Description |
|---|---|---|
| Maximum Object Confidence | 1000 | The maximum value for the object's existence confidence. |
| Minimum Object Confidence | -1000 | The minimum value for the object's existence confidence. |
| Save Object Confidence | 600 | The confidence value at which an object will be stored in the WMKS. |
| Delete Object Confidence | -600 | The confidence value at which an object will be removed from the SLAM+DATMO cache if it is not stored in the WMKS. |
| Initial Object Confidence | 300 | The initial existence confidence of a detected object. |
| Step Detected Confidence | 50 | The step value of the existence confidence when an object is detected. |
| Step Occluded Confidence | -10 | The step value of the existence confidence when an object is found to be occluded. |
| Step Missing Confidence | -50 | The step value of the existence confidence when an object is not detected and is not occluded. |
| Minimum Occluded Confidence | 0 | The minimum value the existence confidence can become when an object is occluded. |
| Free Space Overlap Threshold | 50 % | The percentage of overlap between the free space polygon and the object enclosure that must occur for an object to be considered in free-space. |
| Moving Object Enclosure Overlap Threshold | 10 % | The percentage of overlap between the free space polygon and the object enclosure that must occur for an object to be considered moving |
| Moving Object Segment Overlap Threshold | 0 % | The percentage of overlap between the free space polygon and the object line segments that must occur for an object to be considered moving. |
| Minimum Moving Confidence | -100 | The minimum value for the object's moving confidence. |
| Maximum Moving | 100 | The maximum value for the object's moving |

| | | |
|---|---|---|
| Confidence | | confidence. |
| Initial Moving Confidence | 0 | The initial moving confidence of a detected object. |
| Step Moving Confidence | 10 | The step value of the moving confidence when an object is detected to be moving. |
| Step Static Confidence | -10 | The step value of the moving confidence when an object is detected to be static. |

Table **Error! Use the Home tab to apply Heading 1,00 Chapter Number to the text that you want to appear here.**-6. Continued.

| Parameter Name | Value | Description |
|---|---|---|
| Save Moving Confidence | -30 | The moving confidence value at which an object can be added to the WMKS. |
| Object Update Count | 10 | The number of times an object must have been updated before it will be updated in the WMKS. |
| Object Update Time | 1.0 second | The time between updates of a single object in the WMKS. |

# CHAPTER 4
# TESTING METHODOLOGY

Informal testing was performed throughout the development process in order to ensure the correctness and validity of the implemented algorithms. Testing approaches such as unit tests, integration tests, and regression tests were commonly run. However, a formalized test plan is necessary to evaluate the final performance of the overall system and provide a framework for analysis. In this chapter, the testing methodology used in the completed system is outlined. The chapter begins with a description of the platform used for generating the test data and is followed by an outline of the test plan. Finally, the metrics used for evaluation are presented.

## Test Platform

The Urban NaviGator (Figure ) is an autonomous sports utility vehicle that was developed by CIMAR for the 2007 DARPA Urban Challenge and served as the platform for collecting the data for developing and testing the SLAM+DATMO system presented. It was built off of a 2006 Toyota Highlander Hybrid chassis which was modified significantly to provide the functionality required for autonomous navigation and obstacle avoidance. A hybrid system was chosen to exploit the internal electrical system and reduce the workload required to implement a power management system. In this section the hardware and software details of the test platform will be provided.

### Hardware

**Actuation.** The actuation required for vehicle control was implemented using two methods. Steering and shifting control was implemented using Animatics SmartMotors which were connected to the steering column and the gear shifter respectively. Throttle and brake control was enabled through the existing vehicle drive-by-wire system via a

custom controller that passed “fake” throttle and brake data to the vehicle. Motor position commands and desired throttle and brake efforts were sent to their respective controllers via an attached tablet computer.

**Sensor package.** The Urban NaviGator contains a comprehensive sensor package to provide data for localization, terrain estimation, and obstacle detection. Localization was accomplished by combining data from three GPS units, a GE Aviation North Finding Module (NFM) and wheel encoders using a Kalman filter to provide estimates of the vehicle position and orientation in the global frame. Six SICK LMS-291 and two SICK LD-LRS1000 laser range finders are mounted on the vehicle and are used primarily for terrain estimation and obstacle detection. There are also four Matrix Vision BlueFox cameras mounted to the vehicle which are used for path-finding and lane line detection. Although, there are a myriad of sensors present on the platform, only the localization system and the two SICK LD-LRS1000 range finders were used for the presented research. The fields of view of the two SICK LADAR are shown in Figure 4-1.

**Computing resources.** In addition to the tablet computer required for actuation control, a distributed computing package was implemented. The rear row of seats was replaced with a custom computer rack that could support up to twelve ATX motherboards. Each motherboard contained AMD X2 4600 processors and eighty gigabyte hard drives. The computers ran a mixture of Ubuntu 8.04 linux and Windows XP operating systems and were connected using a gigabit Ethernet network. An in-car development environment was provided using a dual-head keyboard-video-mouse (KVM) switch which connected the installed computers to either one of two rear-seat

workstations. Remote computer access was also provided using a Cisco Aironet 350 Series wireless bridge pair.

**Software**

There are four software elements required for the presented research: the LADAR data server, the Global Positioning (GPOS) component, the WMKS, and the SLAM+DATMO system. The LADAR data server communicates with the SICK LD-LRS1000 range finders over a Controller Area Network (CAN) data interface and rebroadcasts the data to all clients via IP multicast. The use of the data server provides two useful functions. First, it allows multiple software components to use the LADAR data and second, it allows for software to be easily moved from one computer to another without modification to the existing code. The GPOS component provides the position of the vehicle in latitude, longitude, and altitude measurements using the localization sensors described above. Details on the WMKS and the SLAM+DATMO have been previously given. Data exchange between GPOS, the WMKS, and the SLAM+DATMO system was achieved through the use of the Joint Architecture for Unmanned Systems (JAUS) messaging standard.

## Test Plan

Testing was performed using recorded data taken with the Urban NaviGator and reproduced in a custom simulation environment. The LADAR and position data were recorded simultaneously which allowed for the data to be replayed as if happening in real-time or slowed down to facilitate testing. There was no change in the data format between the simulation environment and the actual platform which means the developed algorithms should be able to run on the robotic platform without modification. All testing was conducted under Ubuntu 8.04 LTS on a laptop using an Intel 1.67 GHz

T2300 Duo Core with 4 GB of RAM. Testing was divided into multiple stages with a discussion of each testing stage given below.

### Single LADAR Testing

The first testing stage used a single LADAR to test the different elements of the system. In addition to providing a series of baselines that were used to compare the correctness of the LADAR fusion approach, this testing stage allowed for detecting and fixing simple problems during development and showed “proof of concept” of the presented approaches.

### Static object detection and tracking

First the clustering, line extraction, and static object matching algorithms were tested to evaluate correctness and repeatability. It was important that the clustering, line extraction, and matching algorithms produce repeatable results or the system output could vary widely between tests. Therefore, they were first tested in an environment in which both the vehicle and all objects were static. However, the effect of sensor noise in the LADAR data influences the repeatability of the algorithms. Therefore, the first series of tests fixed the LADAR data fed to the system. After these algorithms were evaluated sensor noise was introduced while still maintaining a static platform and environment. This allowed testing of the object update algorithms as sensor noise would cause an object to be detected differently during each scan. Finally, the influence of platform motion on static object detection and tracking was evaluated by moving the vehicle through a static environment.

### Moving object detection and tracking

Next, the moving object detection and tracking algorithms were evaluated. The first step involved testing the free space violation method to determine its effectiveness

at detecting moving objects. Also, the ability of the system to track the objects over time was evaluated. In order to remove the influence of platform motion on the test results, testing was performed with the platform at a fixed position and with static and moving objects in the environment. Once, the effectiveness of the moving object detection and tracking system was evaluated testing was done with the vehicle moving through a dynamic environment to observe the effect of platform motion on the algorithms.

**Position estimation**

All previous testing was performed without using the position estimation algorithm in order to provide some evaluation of the difference between using and not using position estimation in the system. Therefore, during this testing stage many of the same tests that were previously performed were repeated. First the accuracy of the position estimation system was evaluated with the platform in a fixed position and with no moving objects in the environment. Testing without sensor noise was performed to provide a baseline of the position estimation algorithm followed by testing in the presence of sensor noise to evaluate the influence of sensor noise on accuracy. Next, the LADAR data was fixed while the GPOS provided position was updated as if the vehicle was moving through the environment. This test evaluated the position estimation algorithms accuracy when a known offset was applied. The estimation algorithms were also tested by moving the vehicle through a static environment and checking that it could correctly track the GPOS estimate and make corrections as necessary. Finally, the algorithms were tested by keeping the vehicle in a fixed location and introducing a moving object in the environment to evaluate the effect on the position estimation system.

### World Model Knowledge Store access without position estimation

The next set of testing involved evaluating the ability of the system to add and update objects in the WMKS and testing its ability to identify WMKS objects that no longer exist in the environment. This testing was broken down into three stages. First, the vehicle and environment were both held static while objects were detected and added to the WMKS. The objects stored in the WMKS were then examined for correctness. Next, some objects were modified and new objects were introduced to evaluate the system's ability to not only retrieve stored objects at startup but to modify existing WMKS objects and add new objects to an existing database. Next, the performance of the adding and modifying objects in the WMKS was evaluated when the platform moved through a static environment. This test was performed to ensure that objects could be successfully updated in the WMKS as their representations were modified through platform motion and the effect of the platform motion on retrieving objects from the WMKS. Finally, testing was performed to evaluate the accuracy of adding static objects in the presence of moving objects. It was decided that only static objects would be stored in the WMKS to reduce research complexity. The system assumed that all objects from the WMKS were static and therefore it was important that moving objects were not added.

### World Model Knowledge Store access with position estimation

When objects are added to the WMKS their points are represented in the global frame. However, due to error in GPS, there is no guarantee that if a vehicle is placed in the same position numerous times, the global position will always be exactly the same. One aspect of the system presented is that it should be able to correct for differences between the position of stored static objects and the position of the objects sensed in

the environment. This stage of testing evaluated the capability of the system to do just that. Testing was performed in two stages. First, both the vehicle and the environment was held static with the stored WMKS objects being slightly offset from the sensed objects due to GPOS differences. Next, the vehicle was moved through the static environment. The ability of the system to correctly calculate the position error and maintain the corrected position was evaluated.

### Multiple LADAR Testing

After testing with a single LADAR was completed, a number of tests were performed to evaluate the fusion scheme presented. The effectiveness of the fusion scheme was compared to the results when using a single LADAR. Not all tests that were run with a single LADAR were re-run using multiple LADAR as the exact same algorithms were used in both cases. The purpose of this testing stage was to evaluate the fusion schemes effectiveness without having to modify the algorithms.

## Metrics

A number of metrics were used to evaluate the performance of the system at each stage of testing and to identify areas for improvement. In general, the performance of detection systems have been evaluated by comparing the system output from the expected results of a human. The same approach was taken in this research. After running the number of objects generated by the system were counted and compared to the number of the objects that would be expected by a human. The same approach was performed for both the static and moving object detection systems and provides a general qualitative feel for the accuracy of the system. The time taken for the algorithm to execute was also evaluated and analyzed and the average time for each function logged. Finally, an analysis on the position estimation system performance was done by

tracking the changes of the corrected and uncorrected position estimates from the vehicle origin.

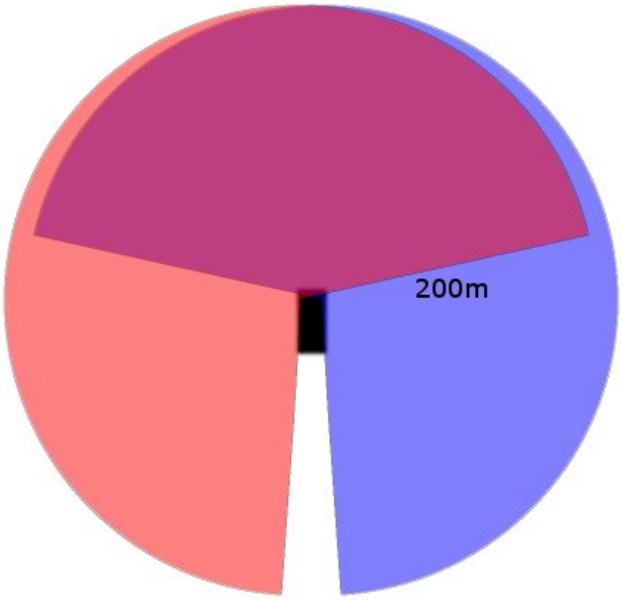



Figure 4-1. Scan regions for the two SICK LD-LRS1000 laser range finders used for testing. The vehicle is represented as the black rectangle in the center of the image.

# CHAPTER 5
# RESULTS

In this section, the results generated from the test plan laid out above will be discussed. Test data was taken from the Gainesville Raceway located in northwest Gainesville, FL. In addition to having a large open area with a number of roads that simulate the road network of an urban environment, the site is closed off from the general public and allows for controlling the presence of moving objects. Although there are a limited number of buildings, the site was more than adequate for use in the required testing. The results from each stage of testing outlined in the test plan from Chapter 4 will be presented and discussed below.

## Single LADAR Testing

As mentioned, the first testing stage involved using a single LADAR to provide a series of baselines. Single LADAR testing was performed using both the driver and passenger side LADARs individually to check if there were any major differences in algorithm performance between LADAR. Figures 5-1 show satellite images of portions of the test site used during testing with overlays of actual point data.

### Static Object Detection and Tracking

Figure 5-2 shows the output from the object detection system when given a set of raw data points. The system is able to successfully detect objects from the scan points and when run without the presence of sensor noise, the system will consistently detect the same objects. Figure 5-3 shows an example of the objects detected from the system using the passenger side LADAR with a static environment and a static vehicle without the presence of senor noise. In Table 5-1, which is based on the results seen in Figure 5-3, it is shown that five static objects were expected while nine objects were detected.

Although it appears that the system detected many more objects than expected, objects 9, 10, and 11 are generated from a single chain-link fence. It is expected that some LADAR beams went through the holes in the fence and caused the incorrect object separation. Also, the detection of objects 3 and 4 are caused by incorrect sensor placement which causes the sensor to hit the ground. When taking into consideration these factors, the performance of the detection algorithm is actually quite good. Object matching without sensor noise was found to consistently match the correct stored and new objects. In general, the system performs well without the presence of sensor noise. Figure 5-4 shows the execution times for running the algorithm without the presence of sensor noise in a static environment with a static vehicle and without the use of the position estimation algorithm or the knowledge store. The average time taken for execution is about 0.13 seconds which is close to real time given that the LADAR collect data at 10 Hz. However, this represents the best case run time for the system and indicates that real time execution of the complete system is not possible with the current algorithm and with computing resources used for testing. Figure 5-5 shows the average time taken for different functions within the system to execute. It can be seen that point association takes the longest amount of time, which is a reasonable result. The algorithm is very inefficient and requires every pseudo cluster point generated from the stored object is compared to every scan point from its matching new object. All other functions that take a long time to execute involve the use of the GEOS library either to generate polygons or to test for polygon overlap.

When sensor noise is introduced to the detection system, the number and shape of the extracted objects starts to vary over time. Figure 5-6 shows how the detection

process is affected over each time step and illustrates the ability of the matching and updating algorithms to correctly identity objects that are the same and update them accordingly. Figure 5-7 shows how objects can be merged over time based on the changes in the sensor data to produce more accurate results. In Figure 5-7A three objects were detected when a single object was expected. However, after some time in Figure 5-7E the three objects were merged into the single expected object. As shown in Table 5-2, which is based on the results shown in Figure 5-8, the performance of the system improves over time due to the influence of sensor noise. Figure 5-9 shows that the introduction of sensor noise has an effect on the overall execution time of the system with the average run time increasing to about 0.19 seconds. Figure 5-10 shows that the average execution time for associating the stored and new object points almost doubles. This increase can be explained by the fact that an old object is updated multiple times when there are multiple matching new objects. When sensor noise was not present, every stored object matched only one new object but now the possibility of multiple matches exists. Therefore, the number of times the point association algorithm is executed is increased. It can also be observed that the time for updating of the existing confidence of the static objects now appears on the graph. This is also reasonable result as some static objects will now overlap with the free space region polygon. In order to determine if an object is free space or not the area of the overlapping polygon is calculated using the GEOS library which is relatively slow.

The next step of testing involved moving the vehicle through a static environment and observing the correctness of the algorithms. It was found that the object tracking and updating algorithms worked well but there were a number of issues. First, the affect

of vehicle pitch and roll greatly affected what objects could be detected by the LADAR. When turning, the roll of the vehicle would cause the plane of the LADAR scan to go above some objects. In these cases, the assumption that the scanning plane of the LADAR was parallel to the ground no longer held and caused a failure of the tracking algorithm. Also, there were inconsistencies introduced into the object representation by the object resolution algorithm. Sometimes, a stored point would be kept when it should have been removed from the representation and caused the object shape to become distorted. Figure 5-11 and Figure 5-12 show a few images of the stored objects when the vehicle moves through the environment. Although, the updating algorithm was sometimes inconsistent it also produced some promising results. It can be seen that the algorithm does successfully outline some of the objects in the environment. One thing that is observed in Figure 5-12 is that the motion of the platform affects the moving object classification process as two static objects were incorrectly detected as moving. This may be caused by errors in the GPOS position estimate since the current testing does not include the position estimation algorithm. Figure 5-13 shows that there is a large effect on the total execution time when the vehicle starts to move through the environment with the average execution time of the system increasing to about 0.35 seconds. In Figure 5-14 it is seen that the main functions that cause this increased execution time are separating the static and moving objects and the updating of the existence confidence of the static objects. Both these functions rely on the GEOS library. Also, platform motion probably causes a lot more objects to be considered during these two functions and therefore would increase processing time.

**Moving Object Detection and Tracking**

Next, testing was done to evaluate the system performance at detecting moving objects in the environment. First, the moving object detection and tracking system was tested with the vehicle in a fixed position in order to avoid any issues that may occur due to platform motion. Figure 5-15 shows a moving object being successfully detected when it comes into view from being occluded. The object is first detected as a static object but is converted to a moving object when the moving confidence passes the threshold. In Figure 5-15D it can be seen that the bounding box is larger than the detected points since the original bounding box would be smaller than the minimum size. However, in Figure 5-15E the length of the bounding box is just about the correct size for the detected points. Figure 5-16 shows an object that starts from a static position and then begins to move. The object has a high static probability which takes a long time to reverse. However, due to the occlusion, the object is split in two and the new half of object is quickly determined to be a moving. As the object moves away from its original position the existence confidence of the remaining misclassified static object goes to the minimum indicating that it is missing. When a moving object is successfully detected the number of tracking errors due to partial occlusion is decreased. In Figure 5-17 the object moves behind a static object and becomes partially occluded. However, the system is able to successfully and correctly track the object until it is no longer occluded. However, if the object shape changes drastically and does not correspond to the expected ‘L’ shape of a car or truck, the moving object tracking algorithm breaks down. Figure 5-18 shows the failure of the tracking algorithm, due to a bad sensor position, which causes the LADAR to move from striking the side of the vehicle to striking its wheels. When this occurs it can be seen that the bounding box does look as

expected. Also, sporadic static objects start to appear as the new clusters can no longer be associated with the moving object. The total execution time for the system during this testing stage is shown in Figure 5-19 and is found to be similar to the execution times with a static platform and a static environment. The average execution time is about 0.20 seconds. However, the function that causes the greatest slow down is seen to be function that separates the objects into moving and static objects instead of the point association function (Figure 5-20).

The final step in testing the detection and tracking systems involved testing with a dynamic platform in a dynamic environment. This test was used to evaluate the completeness of the system. Upon examination of Figure 5-21 it can be seen that the algorithms were successful in tracking the moving object as the platform moved through the environment. The system was also able to simultaneously track the moving object and update the static object representations. The problems of inconsistent static object updating and incorrect classification of static objects as moving objects (Figure 5-22) that were seen when the platform moved through a static environment were also seen during this stage. The average execution time for the algorithm during this stage of testing was found to be about 0.22 seconds (Figure 5-23) with the point association and object separation functions taking the longest time (Figure 5-24).

**Position Estimation**

The evaluation of the position estimation system was done in five stages. First, the system was evaluated with fixed LADAR data and a static platform and static environment. This test provided a baseline for comparison of the position estimation performance in all other testing stages. Figure 5-25 shows how the distance from the origin changes over time for the corrected and uncorrected position estimates. First, it

should be noted that the uncorrected position estimate does change over time although the vehicle is not moving due to drift in the GPOS position. Also, it is seen that the corrected position does not exactly follow the uncorrected position. This is due to the introduction of error by the object approximation and association process. Although, the LADAR points are the same during every cycle the pseudo points used during association are generated from simplified object representations. Therefore, it is impossible for the old points and the new points to ever be perfectly aligned and introduces an inherent error in the position estimate. However, the error introduced is small. Maximum errors of 0.14 m and 0.10 m are seen in the UTM x and y directions respectively and a rotational error of less than 0.30 degrees.

When real LADAR data was used sensor noise had an effect on the position estimate as can be seen in Figure 5-26. In addition to the error introduced by the object approximation described above, the LADAR data itself changes over time and introduces error. However, the error between the corrected position and the origin is still relatively small. Errors of 0.20 m or less are seen in the UTM x and y directions and a rotational error of less than 1 degree is experienced.

One interesting test performed was to change GPOS position estimate in a known and repeatable manner but fix the LADAR data. Ideally, the corrected position estimate should remain unchanged while the uncorrected position estimate follows he vehicle motion. Figure 5-27 shows that the position estimation algorithm performed fairly well. The corrected position estimate of both the UTM x and y positions are within 1 m from origin as opposed to an uncorrected UTM y position of over 15 m. Also, the corrected

yaw position is within 2 degrees of the origin versus an uncorrected yaw position of over 25 degrees.

Next, the ability of the corrected position estimate to correctly follow the vehicle as it moves through the environment was evaluated. The platform was moved through a static environment and the corrected and uncorrected position change was monitored. Figure 5-28 shows how the corrected position estimate correlated to the uncorrected position estimate. It can be seen that the corrected and uncorrected position estimates followed each other very closely for most of the graph. However, the corrected and uncorrected position estimates in the UTM y position diverge slightly near to the end of the graph. Unfortunately, it is not possible to determine if this divergence is caused by an error in GPOS as a result of GPS or by the map based correction.

Finally, the position estimation system was tested in the presence of moving objects. In order to evaluate its effectiveness, the vehicle was fixed in place and a moving object was allowed to move through the environment. Figure 5-29 shows how the position estimate was affected by the presence of a dynamic object. In general, the corrected position seems to behave in a similar manner to the previous tests with a static vehicle. Although, the corrections in the UTM y position and rotation appear to be noisier, it is unclear if this was caused by the presence of the moving object or not. However, the error of the system is still reasonable. The change in the UTM x and y positions are less than 0.30 m and 0.15 m respectively and the change in the yaw is less than 0.5 degrees.

**World Model Knowledge Store without Position Estimation**

The next series of tests involved the use of the WMKS and ensured that objects could be successfully stored, updated, and retrieved. Figure 5-30 shows that static

objects were successfully added to the WMKS when the object existence confidence passed the threshold value. Eventually, the previously stored objects 29 and 31 were merged into object 29. Figure 5-31 shows that the updated object 29 is successfully updated in the WMKS and object 31 is no longer visible as its identification field has been changed to DUPLICATE. Next, object retrieval from the WMKS was tested. Objects that were generated using the driver side LADAR and were previously added to the WMKS were retrieved to “seed” the SLAM+DATMO system. The passenger side LADAR was then used to update the local objects and the WMKS was checked to ensure that the previously stored objects were updated and the new objects were added (Figure 5-32 and Figure 5-33). Next, the ability of the system to detect that previously stored objects no longer existed in the environment was evaluated. Object 29 was added to the WMKS and then moved to a new location. As seen in Figure 5-34 the system was able to successfully decrease the existence confidence of object 29 to a value -1000 and increase the confidence on the newly detected object 43 up to a value of 1000 and update their confidence values in the WMKS. Tests were also performed to ensure that the objects were updated in the WMKS as their representation was modified by motion through the environment and that moving objects were not added to the WMKS with the results shown in Figure 5-35 and Figure 5-36.

**World Model Knowledge Store with Position Estimation**

When an object is added to the WMKS it is stored in the global frame. However, when the objects are retrieved at a later time, there is no guarantee that the global position will be the same due to the influence of error in GPOS. Figure 5-37 shows a situation where the stored WMKS are not aligned with the current LADAR scan. Therefore, it was important that the vehicle position can be corrected to cause the

stored objects and the current LADAR scan to become aligned. The first test involved checking if the position estimate would correct the error and maintain the correction if the vehicle is not moving and with no moving objects present. Figures 5-38 to 5-40 show the position correction system successfully adjusting the vehicle position to reduce the discrepancy between the WMKS object positions and the sensed object positions. In Figure 5-41, it can be seen that the correction system immediately applies a correction, which is consistent with the difference between the WMKS objects and the sensed objects, and is able to maintain the correction fairly well. When examining the execution times of the entire algorithm it was found that the average execution time was 1.08 seconds (Figure 5-42) with the position estimation algorithm taking an average of about 1 second to run (Figure 5-43).

The position correction between the stored WMKS objects and the current LADAR scan was also tested with the platform moving through the environment to evaluate if the position correction will accurately follow the uncorrected position estimate. Figure 5-44 shows that an initial correction is applied and maintained for most of the graph especially when considering the graph shown in Figure 5-44A. It is interesting to note that the corrected and uncorrected positions start to converge close to the end of the graph. Platform motion through the environment did not greatly affect the execution time of the entire algorithm. The average execution time was found to be about 1 second (Figure 5-45) with the position estimation algorithm again taking an average of about 1 second to run (Figure 5-46).

### Multiple LADAR Testing

The LADAR fusion scheme was tested next. Figure 5-47 shows a satellite image with an overlay of data points from both the LADAR while Figure 5-48 shows a close up

image of the points. It can be seen that the driver and passenger side LADAR points are not aligned which caused a problem when running the system. Due to this large discrepancy in point alignment some of the implemented algorithms produced undesirable but predictable results. Therefore, not all the tests run using a single LADAR were re-run.

**Static Object Detection and Tracking**

The performance of the static object and tracking system was first tested with a static platform in a static environment. Figures 5-49 to 5-501 show some output from the system. It was found that the object detection system worked well without any changes and the objects could be correctly updated. The misalignment of the points between the LADAR had minimal effect in this test and was treated as sensor noise by the system. It was found that the object representations were updated faster than when using a single LADAR. This result was expected since there was now twice as much data and one LADAR would compensate for spaces incorrectly introduced by the other LADAR. The average time take for the system to process a single LADAR was about 0.19 seconds and Figure 5-52 shows the general trend for the total execution time. It can be seen that the algorithm performs in a similar manner to the same test using a single LADAR. One difference between two tests was the increase in average execution time for updating the static object confidence (Figure 5-53). This can be explained by the fact that the scan points are not aligned and therefore, more objects overlap the free space polygon than when a single LADAR is used.

When the system was tested with platform motion through the environment it was found that the misalignment between the scan points became very large (Figure 5-54). This large error between the scan points caused many of the algorithms to break down

as some of the simplifying assumptions about the format of the data no longer held. However, for the small interval when the alignment error was not large enough to cause incorrect object matches, the system seemed to perform fairly reasonably. Due to this misalignment problem any further testing with platform motion through the environment would be inconclusive. Therefore no further tests with a dynamic platform were performed.

**Moving Object Detection and Tracking**

The next step involved evaluating the performance of the fusion scheme when detecting and tracking moving object's in the environment. In general, the system performed better at detecting moving objects when using multiple LADAR than when using a single LADAR. Figure 5-55 shows an object that is detected when it is stationary and begins to move. The system is able to quickly detect the motion and re-classify the object. This result is not surprising as there is now more data available in order to make a better classification decision. However, the tracking algorithm performs worse than when using a single LADAR due to the differences between scan point alignments. Figure 5-56 shows the detection of the object using the driver side and passenger side points and it can be seen that the bounding box does not enclose all the points from both LADAR. Therefore, when the LADAR used is switched the position of the bounding box changes and does not smoothly track the object through the environment.

**Position Estimation**

It was expected that the position correction provided by the system would oscillate due to the scan point misalignment. Figure 5-58 shows that this is indeed what happened. The object detection algorithm averages the object position between the misaligned scan points such that the detected object lies between the two point sets

(Figure 5-57). Therefore, when the position correction is calculated it is oscillates between the positive and negative directions. Due to this oscillation effect no further testing of the position estimation algorithms was deemed useful.

**World Model Knowledge Store without Position Estimation**

Finally, the effect of using multiple LADAR on when objects were added to the WMKS was explored. As mentioned, it was determined that no further testing with platform motion would be useful and since moving objects are not added to the WMKS, testing with the WMKS was only performed with a static platform and static environment. When the same test was performed using a single LADAR, it was found that most detected objects were added to the WMKS at the same time and then updated as necessary. However, when using multiple LADAR some objects were added to the WMKS before other objects. This was due to the difference in the rate of change of the existence confidence of the objects that were detected by both LADAR. Figure 5-59 shows how the existence confidence of the objects can vary greatly due to sensor overlap. Figure 5-60 shows the objects in the WMKS at two different times. The objects that were initially added lie within the overlapping region of both LADAR and therefore, their existence confidence increased quickly. One interesting by-product of the fusion approach was the correction of error from one LADAR by the other LADAR. The placement of the passenger side LADAR caused it to hit the ground and incorrectly detect the ground as a static object. When the passenger side LADAR was tested alone, these “ground objects” were added to the WMKS as is shown in Figure 5-61A. However, since they are not detected by the driver side LADAR, their existence confidence never passed the threshold value required for them to be added to the WMKS when using both LADAR (Figure 5-61B).

## Discussion

A discussion on the results obtained will now be presented. This section seeks to provide the author's assessment of the proposed SLAM+DATMO system and the LADAR fusion scheme and identify weaknesses, shortcomings, and possible future improvements.

### Single LADAR Performance

In general, the system performed fairly well when using a single LADAR. However, performance was greatly improved when the platform was fixed in the environment. The object detection and tracking algorithms were able to correctly identify objects and deal with the presence of sensor noise. The representations of static objects were accurately updated when possible and were not adversely affected when the LADAR data caused the algorithm to incorrectly detect a single object as multiple objects. Moving objects could be detected and consistently tracked as they moved through the environment even when they were partially occluded. Moving object tracking was seen to be fairly smooth despite that fact that a simplistic approach was taken for matching and tracking. The moving objects were only matched based on an overlap method as opposed to the use of Kalman filters or linear predictions as seen in other work. The addition of a more sophisticated matching and tracking algorithm should greatly improve performance. Static objects were correctly added and updated in the WMKS and could be retrieved at a later time. Also, objects in the WMKS could be detected as missing through the use of the existence confidence which would decrease to a minimum value when an object was removed from the environment.

However, when platform motion was introduced, the detection and updating algorithms did not perform as well. Static objects would sometimes be misclassified as

moving and inconsistent object representations would occur. In fact, the update algorithms would sometimes completely fail. One possible reason for object misclassification is the incorrect correlation between the vehicle position and the scan. The free space polygon used to detect moving objects is stored in the global frame, and therefore, the global position of the vehicle needs to be known. However, the LADAR and the positioning system are not synchronized and the exact position of the vehicle when a LADAR scan is taken in unknown. A best guess estimate is used based on the time of the scan and the position of the vehicle at that time. However, this estimate could be incorrect due to lag in the positioning system, lag in the LADAR, or some other error. If this occurs, objects would appear to move when they were in fact stationary.

The position estimation algorithm was also shown to perform reasonably well. The error introduced by the method was small and did not have an adverse effect on the results. The corrected position estimate was able to consistently follow the uncorrected position when platform motion was introduced and kept the vehicle close to the origin when the LADAR data was artificially fixed and the vehicle's GPOS position was allowed to change. It is important to note that the position correction system did not incorporate any form or smoothing, filtering, or averaging in order to improve performance and was based solely on the position of the objects in the environment. The correction between each time step was simply calculated and applied. It is believed that performance will greatly improve if the calculated correction was incorporated into the Kalman filter used in the GPOS component that also incorporates the GPS, NFM, and wheel encoder inputs. One alternative method to the one presented would use the points along the objects extracted in the current scan instead of the raw data. Pseudo

points could be generated from the extracted objects similar to the points generated along the stored object and be used during the point association stage. This method would allow objects to become perfectly aligned when sensor noise is not present.

The biggest weakness of the presented system was the amount of time taken for processing. Real-time operation was not possible on the test hardware especially when position estimation was used. One way to improve the average execution time would be to run the position estimate at a slower rate, such as once per second. However, one important caveat is that the error in the initial estimate must be small to facilitate accurate object matching. If the error is large objects will not be correctly matched and the matching algorithm would have to be repeated which would add additional processing time. Also, optimization of the functions used from the GEOS library and the parallelization of some of the algorithms would greatly improve processing speed.

**LADAR Fusion Scheme**

The presented LADAR scheme worked fairly well with much better performance when the vehicle was static than when the vehicle moved. However, this is not surprising as it was also true of the system performance when using a single LADAR. The use of multiple LADAR improved the performance of the detection algorithms due to increased amount of data and made object tracking more robust against occlusion. However, the scheme was highly dependent on the correct alignment of the two LADAR and could cause system failure if the misalignment was large enough. Although, the expectation of sensor alignment is not unreasonable, the accuracy of the alignment needed for the fusion method should be quantified since exact alignment between different sensors is very difficult if not impossible.

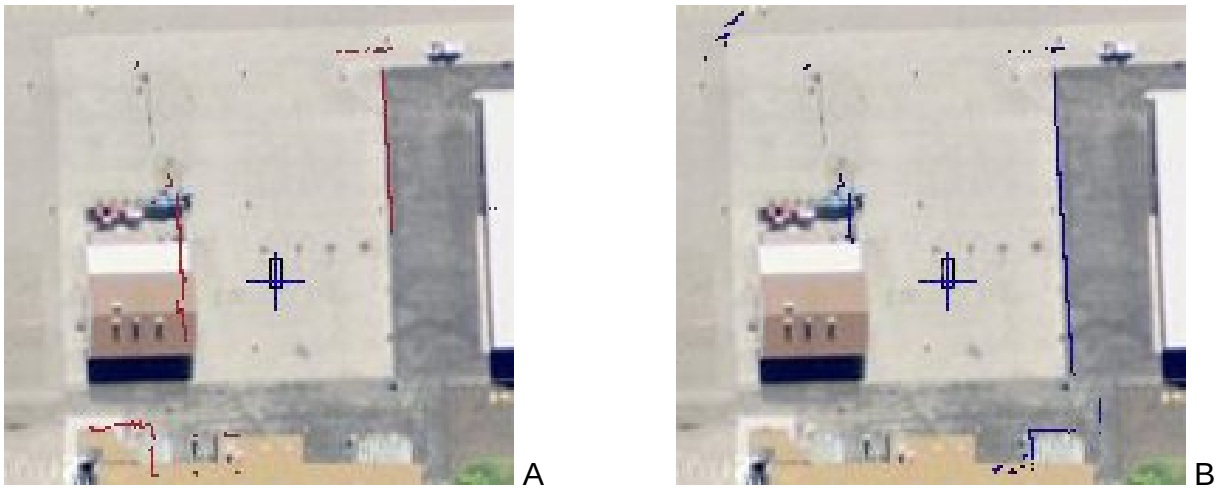



Figure 5-1. Satellite imagery from the Gainesville Raceway with an overlay of LADAR point data. A) The data obtained from the driver side LADAR. B) The data obtained from the passenger side LADAR.

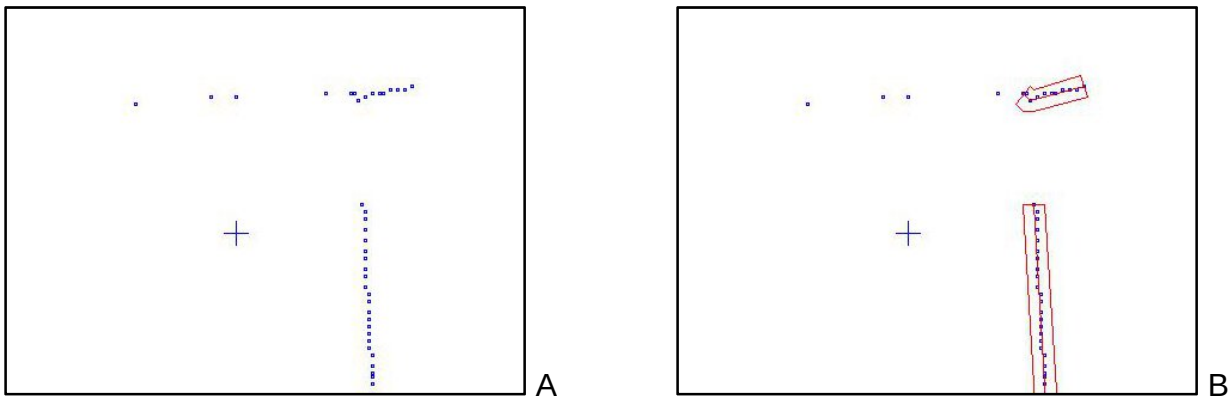



Figure 5-2. Extraction of Objects from Sensor Points. A) The raw scan points obtained from the passenger side LADAR. B) The objects extracted from the raw data with the object enclosures shown after running the detection algorithm.

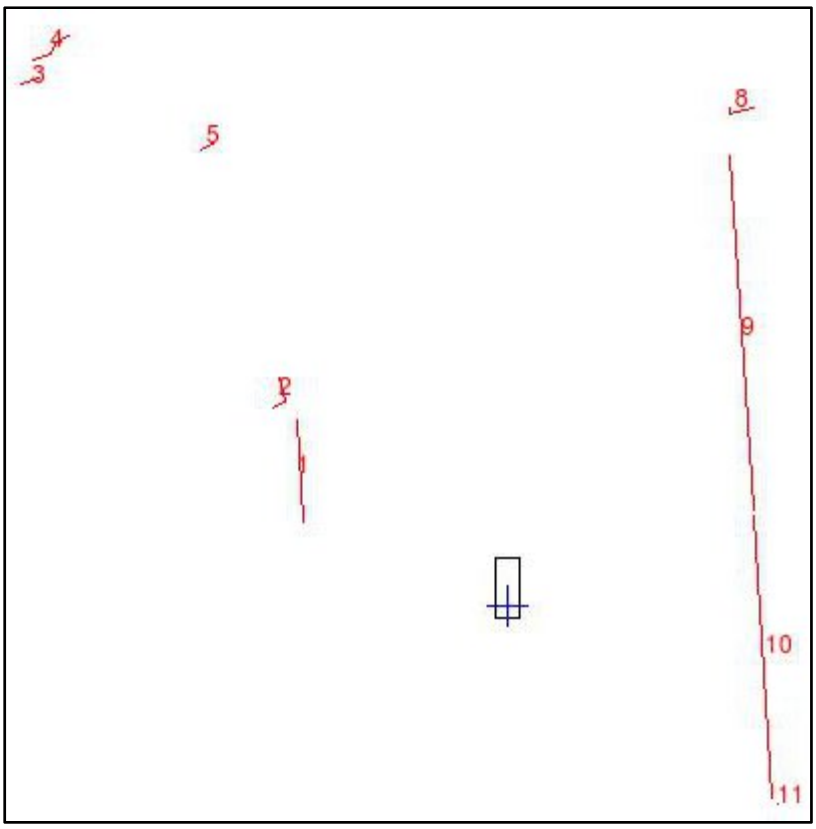



Figure 5-3. Objects detected using data from the passenger side LADAR.

Table 5-1. Expected objects versus detected objects based on Figure 5-3.

| | Expected | Detected |
|---|---|---|
| Static Objects | 5 | 9 |
| Moving Objects | 0 | 0 |

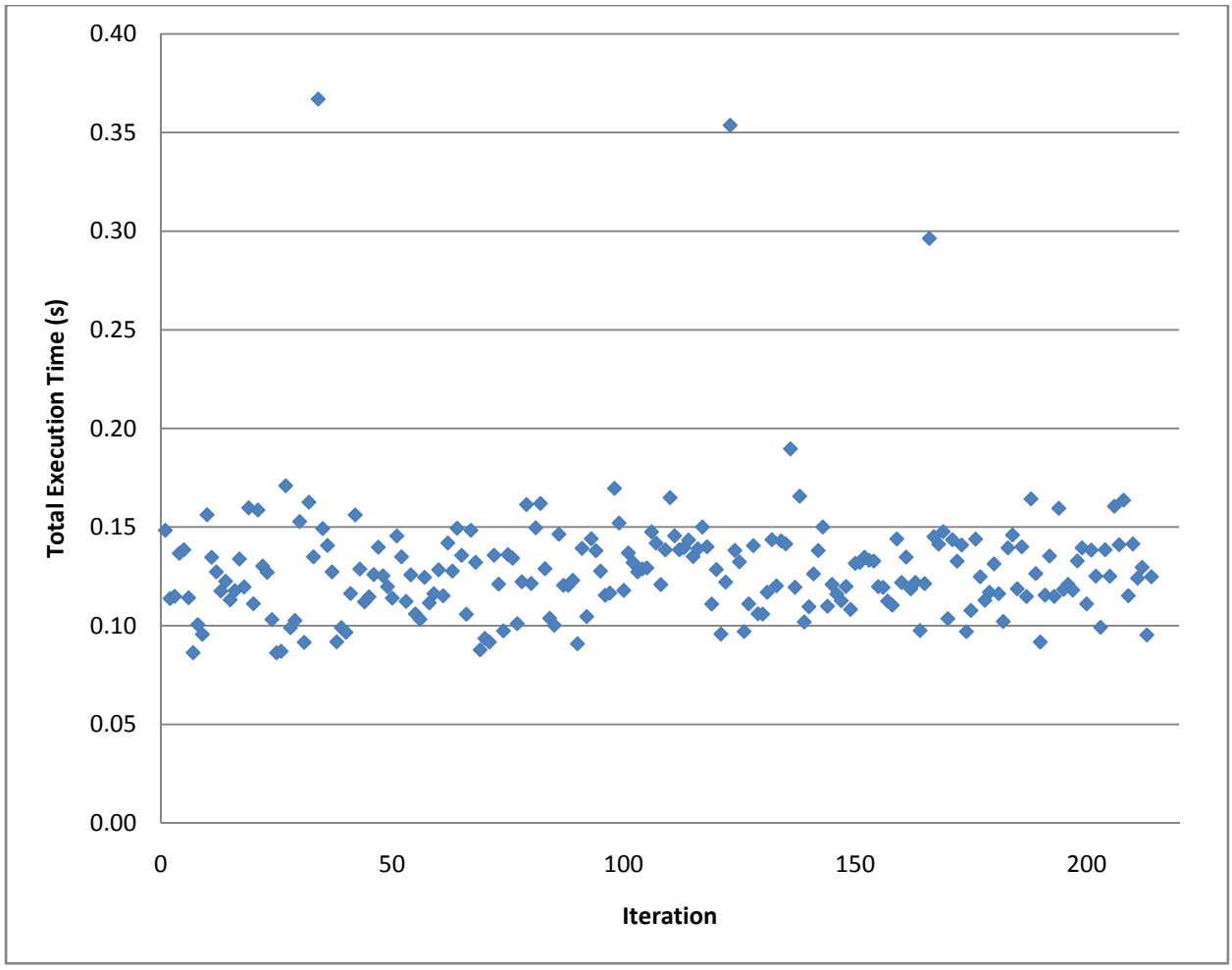



Figure 5-4. Total execution times for the system using the passenger side LADAR with a static vehicle and static environment without the presence of sensor noise, the use of position estimation, or access to the world model knowledge store.

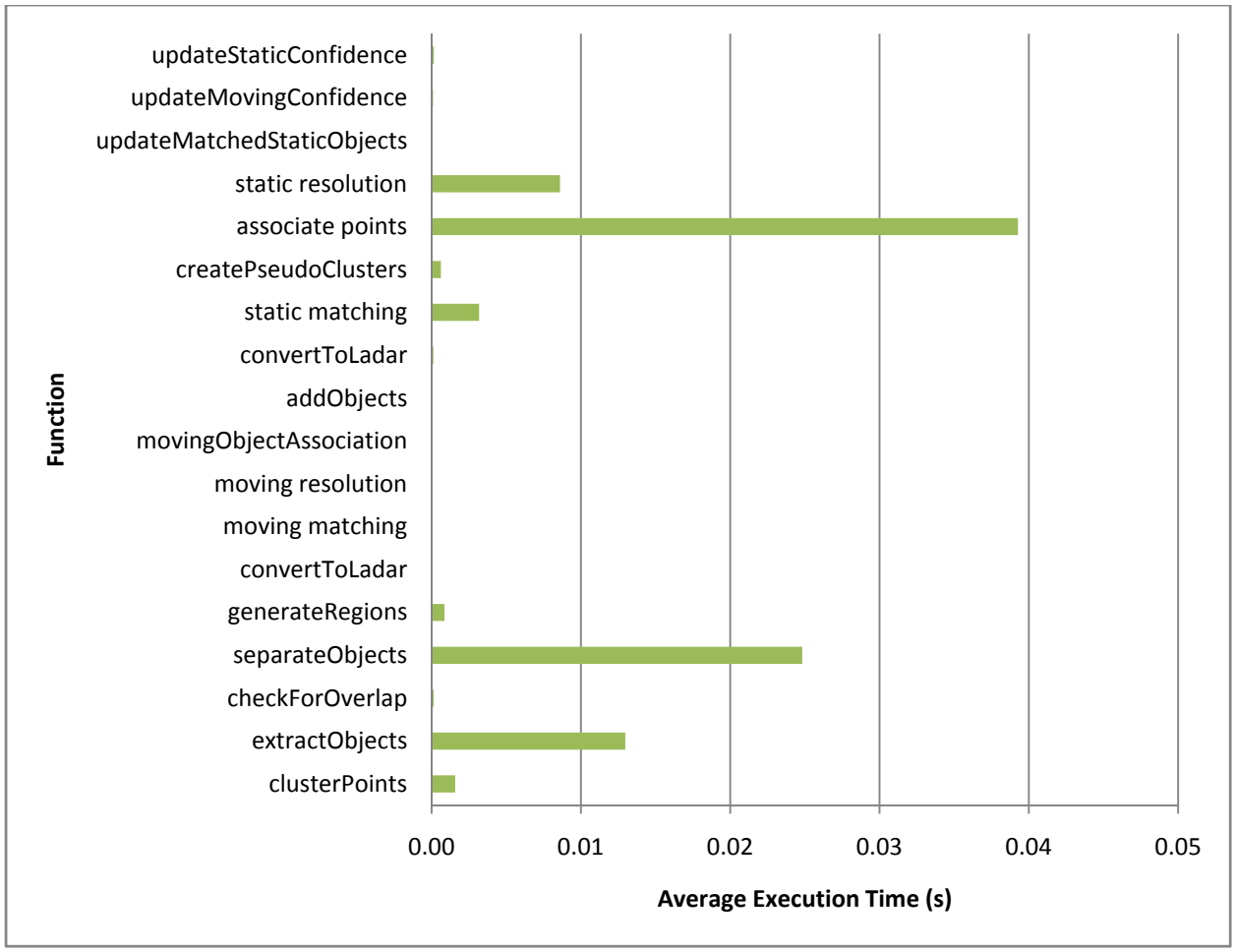



Figure 5-5. Average execution times for different functions using the passenger side LADAR with a static vehicle and static environment without the presence of sensor noise, the use of position estimation, or access to the world model knowledge store.

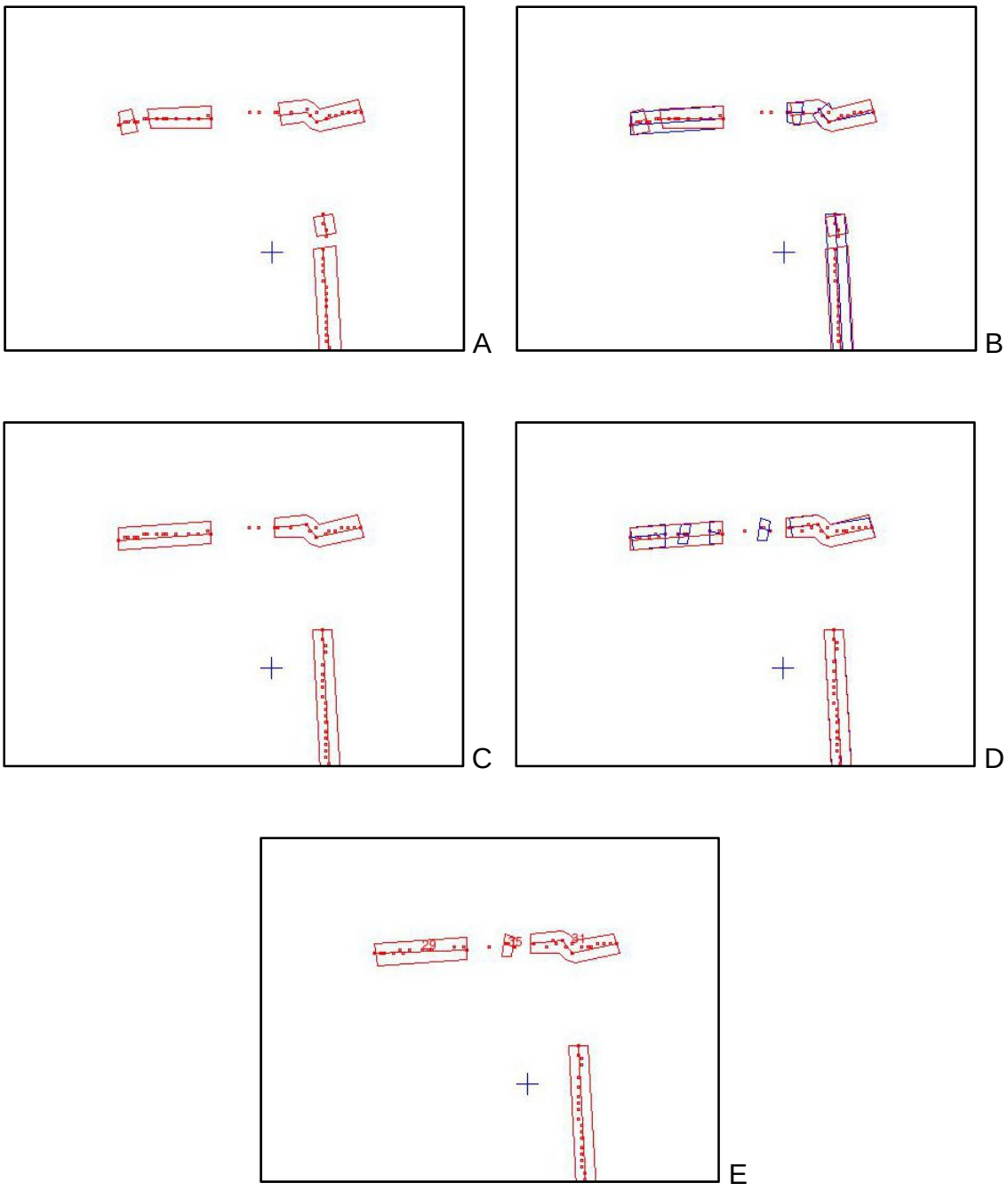



Figure 5-6. Sensor noise causes the extracted objects to vary over time. A) Five objects were detected at t=0 and stored (red). B) Four objects (blue) are detected at t=1 which overlap all the previous objects. C) The update of the stored objects using the new objects caused the number of objects to be reduced to three. D). Six objects are detected at time t=2. E) The number of stored objects increases to four.

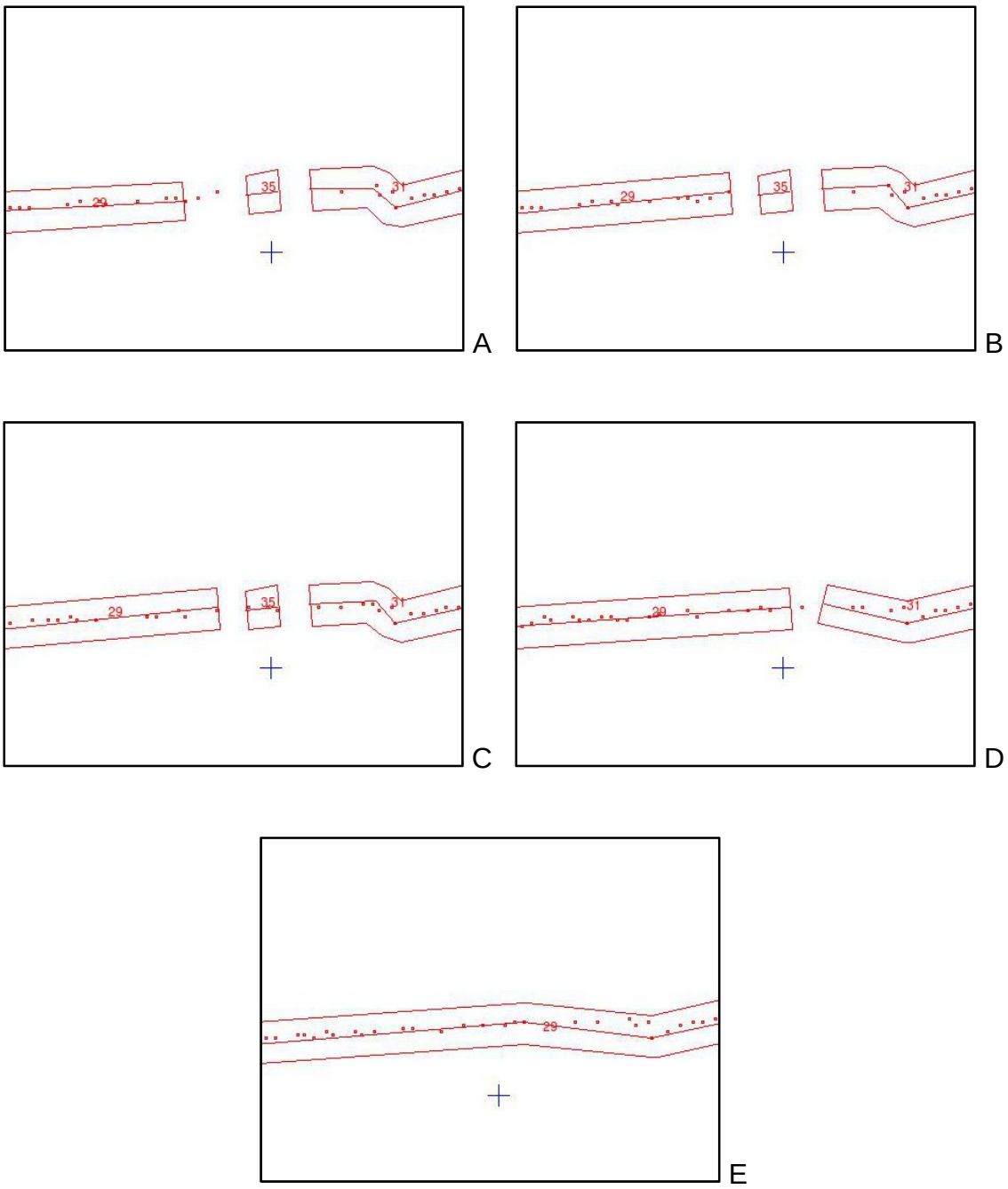



Figure 5-7. The resolution algorithm allows objects to be combined and updated over time. A) Three objects were detected and stored. Although, object 35 has no points associated with it was detected in a previous scan and remains stored. B) Object 29 is extended as the previously unassociated LADAR points shift due to differences between scans. C) The objects do not change despite differences in sensor data as the changes between the points are small. D) Objects 29 and 35 are merged into a single object. E) Objects 29 and 31 are merged.

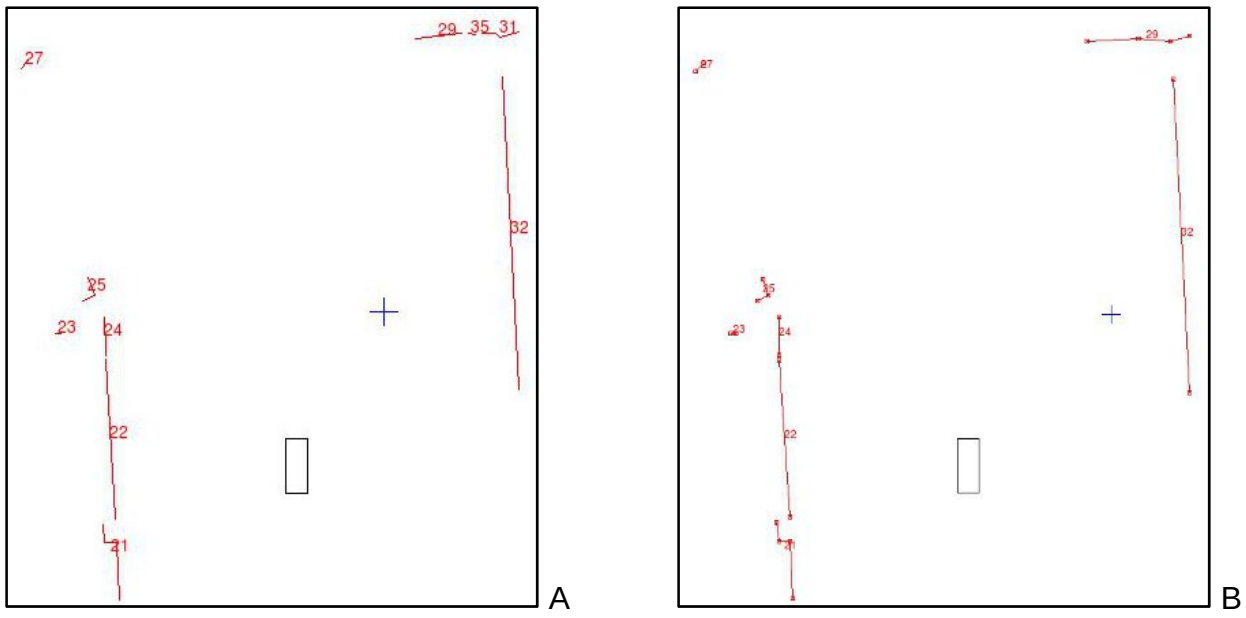



Figure 5-8. Objects detected using data from the driver side LADAR. A) Initial objects detected. B) The objects remaining after some time has elapsed.

Table 5-2. Expected objects versus detected objects based on Figure 5-7.

| | Expected | Detected at A | Detected at B |
|---|---|---|---|
| Static Objects | 5 | 10 | 8 |
| Moving Objects | 0 | 0 | 0 |

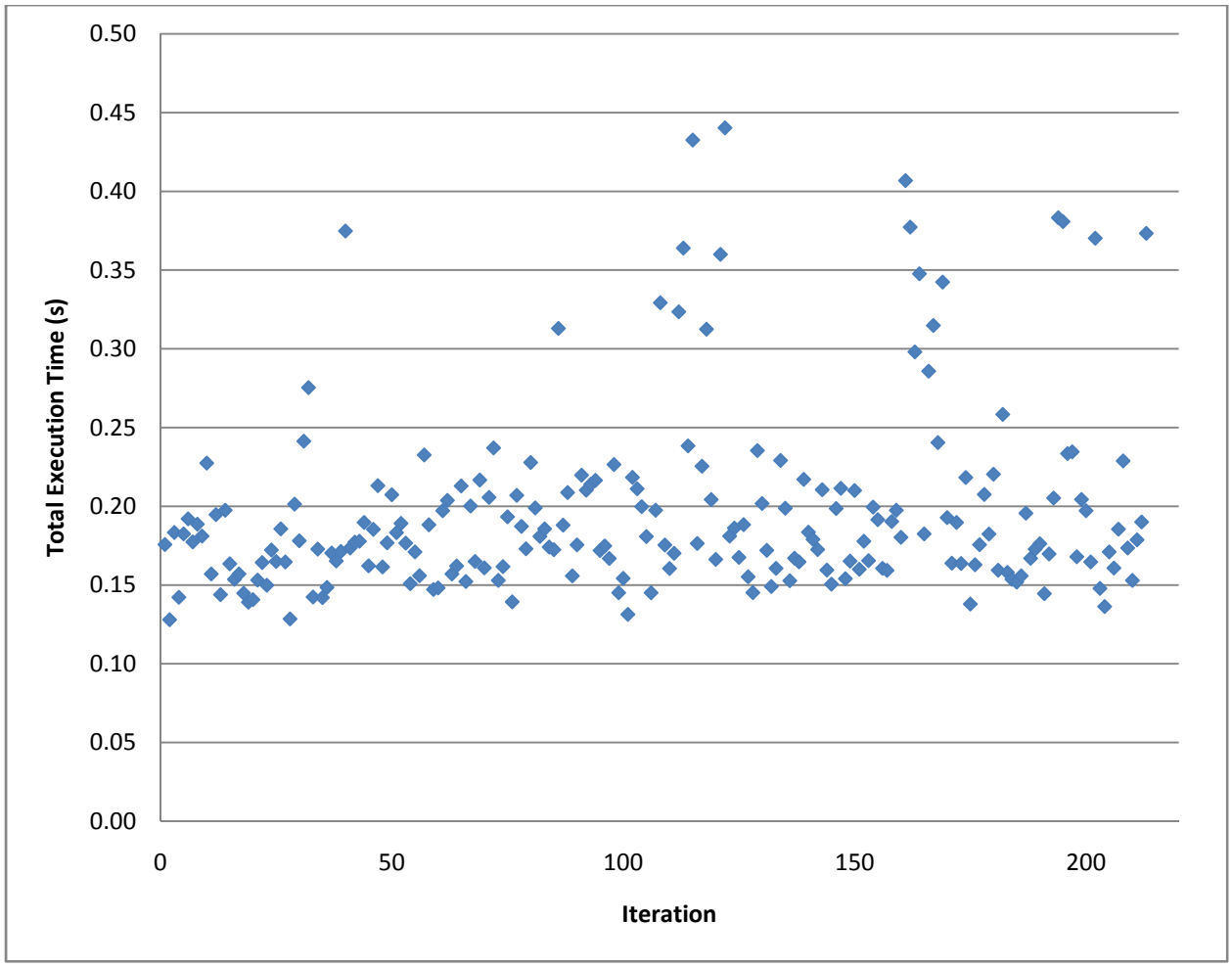



Figure 5-9. Total execution times for the system using the passenger side LADAR with a static vehicle and static environment with the presence of sensor noise but without the use of position estimation, or access to the world model knowledge store.

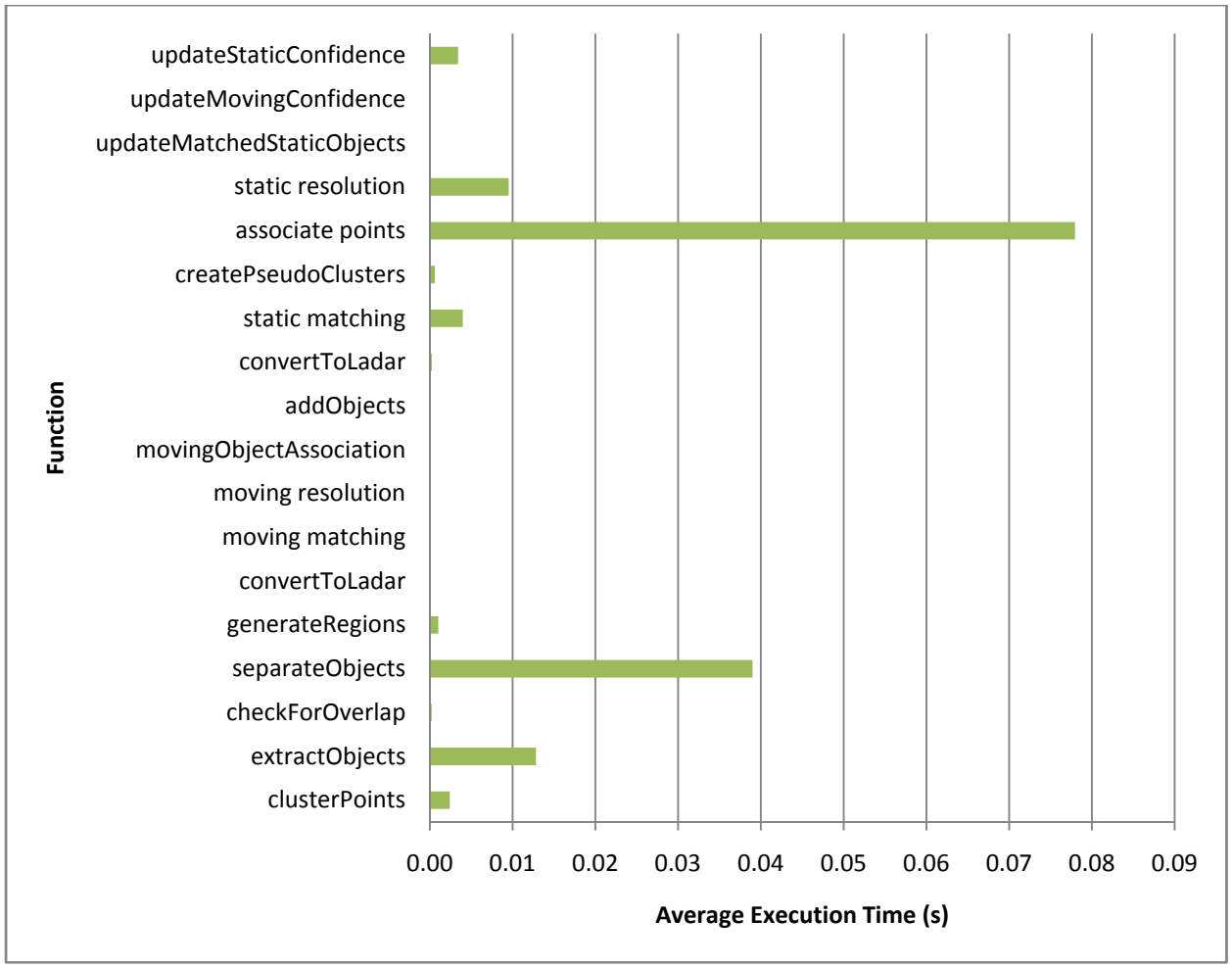



Figure 5-10. Average execution times for different functions using the passenger side LADAR with a static vehicle and static environment with the presence of sensor noise but without the use of position estimation, or access to the world model knowledge store

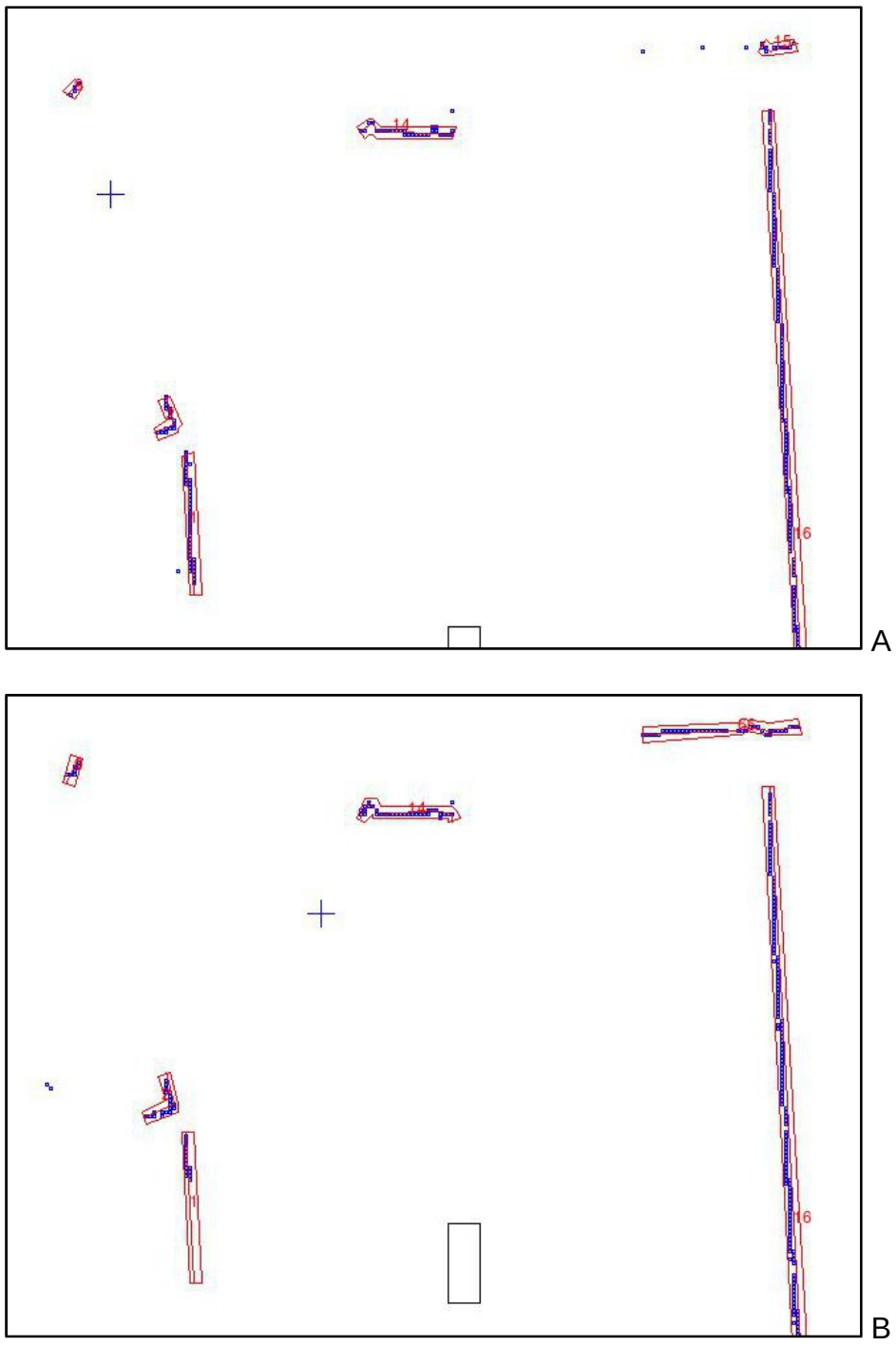
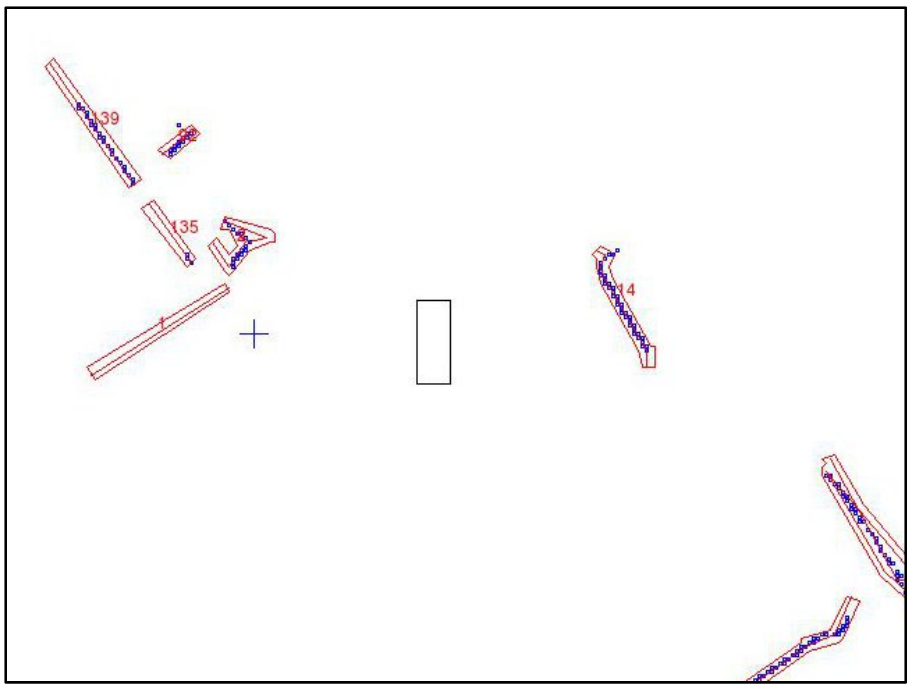
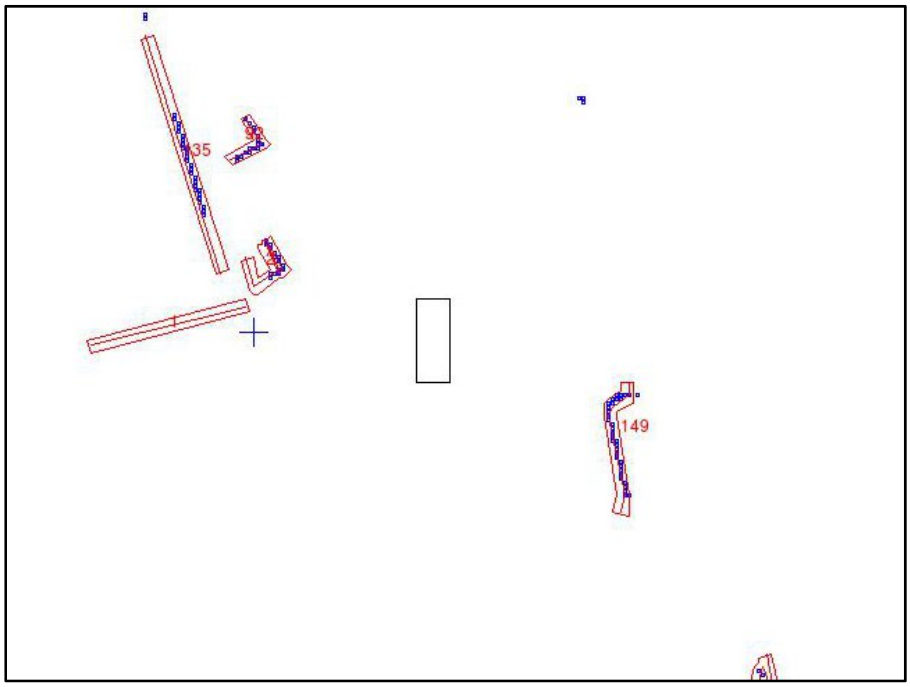



Figure 5-11. Objects detected using the passenger side LADAR when the vehicle moves through a static environment. A) Six objects are initially detected. B) The initial object matching and updating process appears to work well when moving in a straight line. C) As the vehicle makes a corner it is observed that the update algorithm sometimes causes inconsistent representations (object 2 next to object 135). D) The update algorithm is able to successfully merge objects 139 and 135 and produce reasonable outline for object 2.




C




D

Figure 5-11. Continued

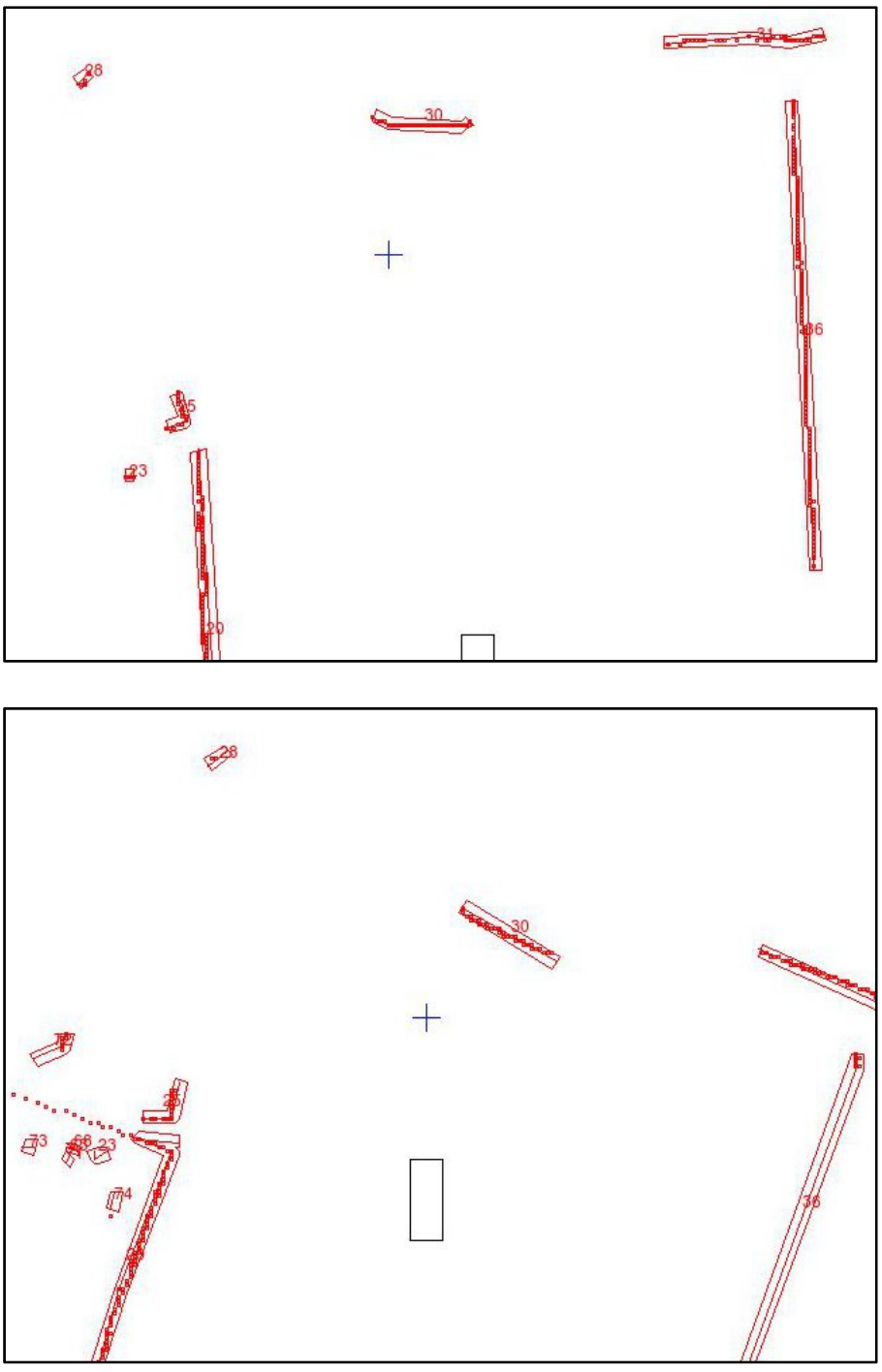
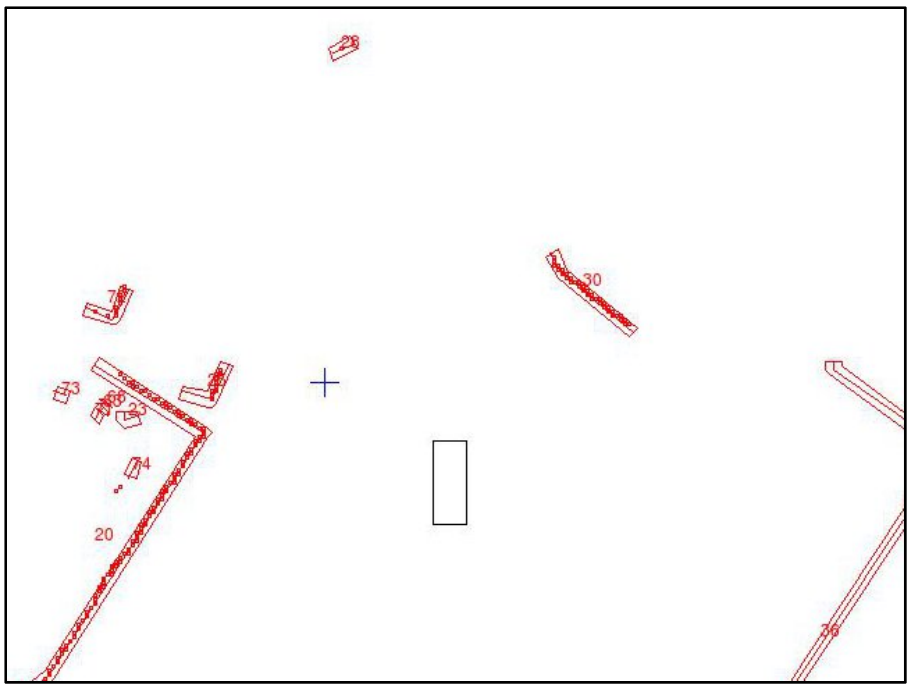
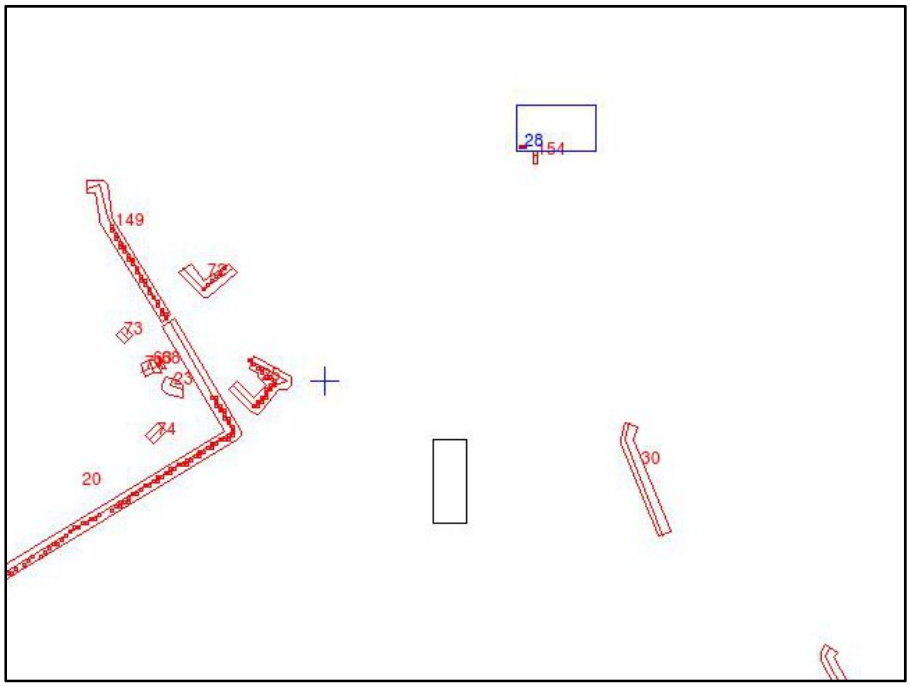



Figure 5-12. Objects detected using the driver side LADAR when the vehicle moves through a static environment. A) Seven objects are initially detected. B) The update algorithm has a problem dealing with the corner of the building (object 20). C) The building corner representation is fixed. D) A moving object is incorrectly detected. E) The building outline (object 20) looks reasonable but the other object representation does not and another moving object is detected.




C




D

Figure 5-12. Continued

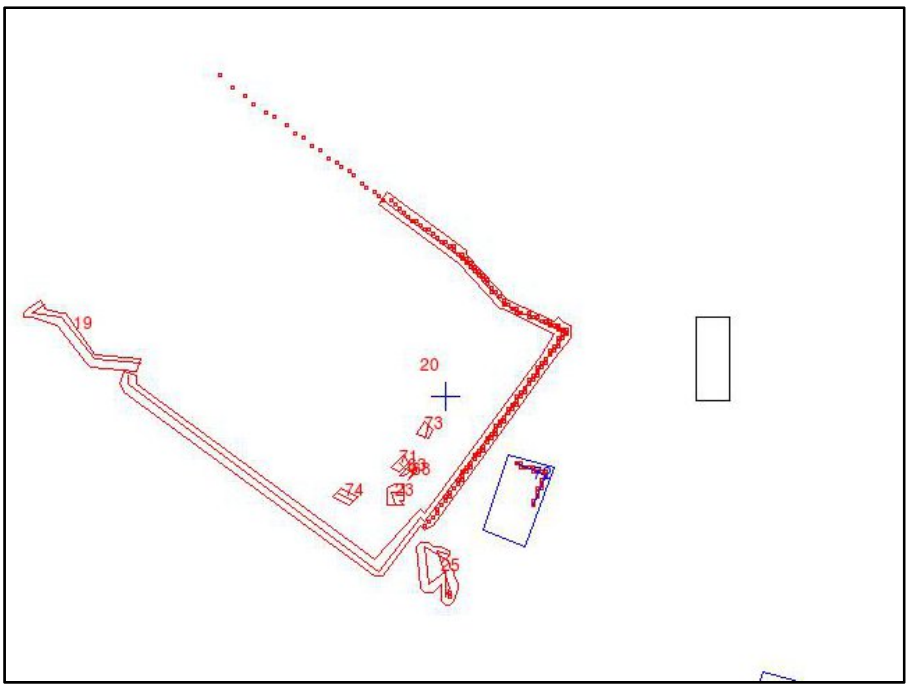



Figure 5-12. Continued

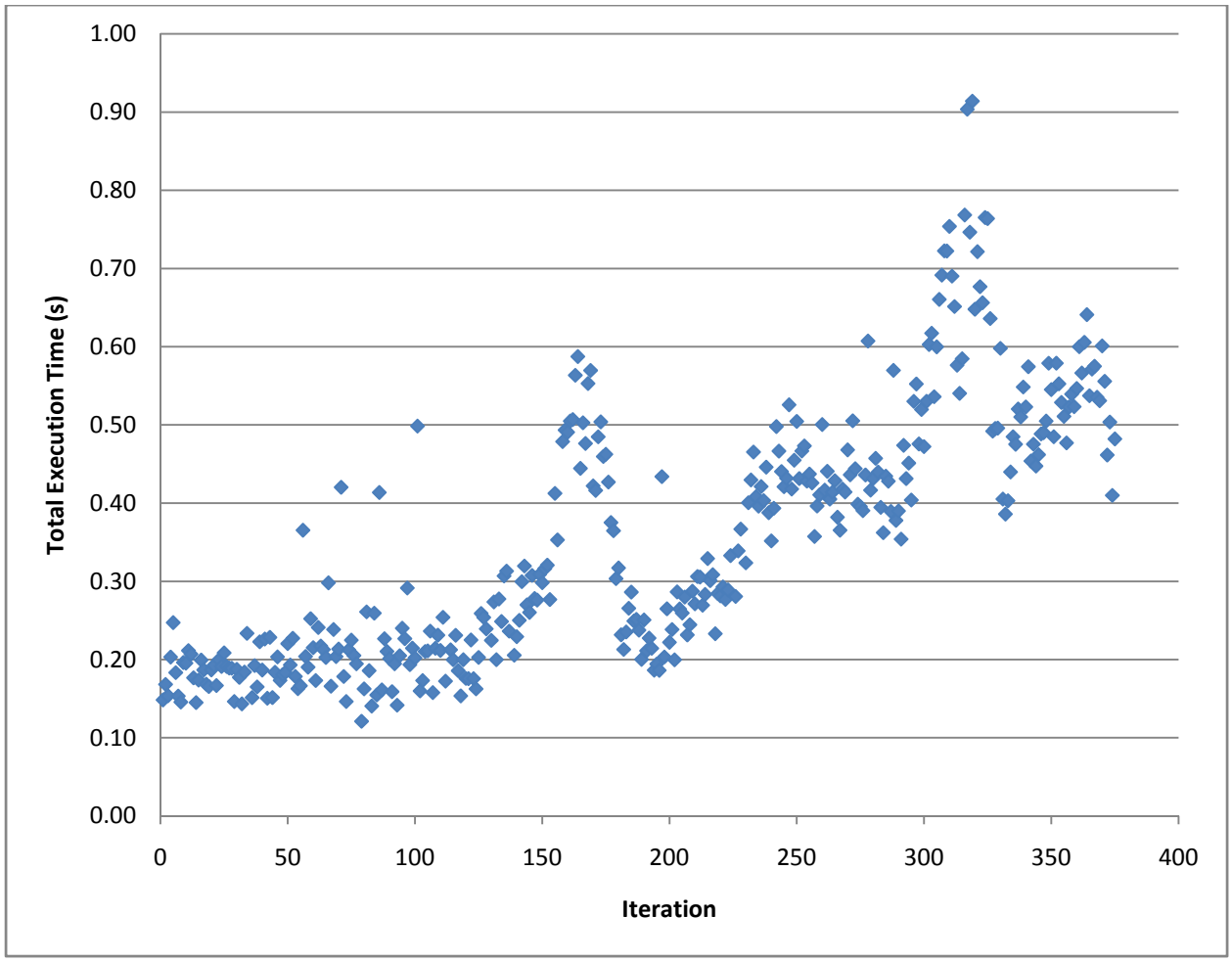



Figure 5-13. Total execution times for the system using the driver side LADAR with a moving vehicle and static environment with the presence of sensor noise but without the use of position estimation, or access to the world model knowledge store.

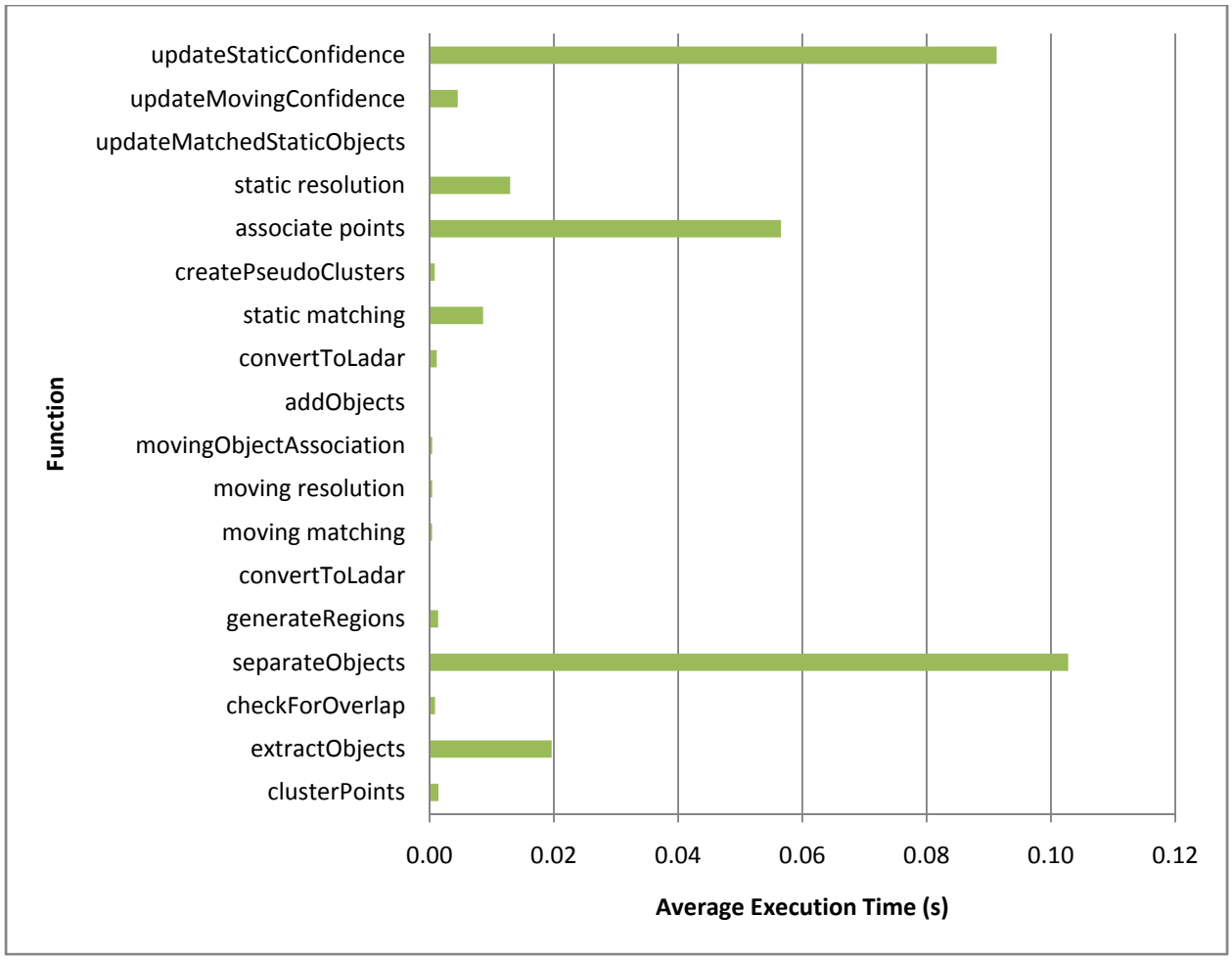



Figure 5-14. Average execution times for different functions using the driver side LADAR with a moving vehicle and static environment with the presence of sensor noise but without the use of position estimation, or access to the world model knowledge store

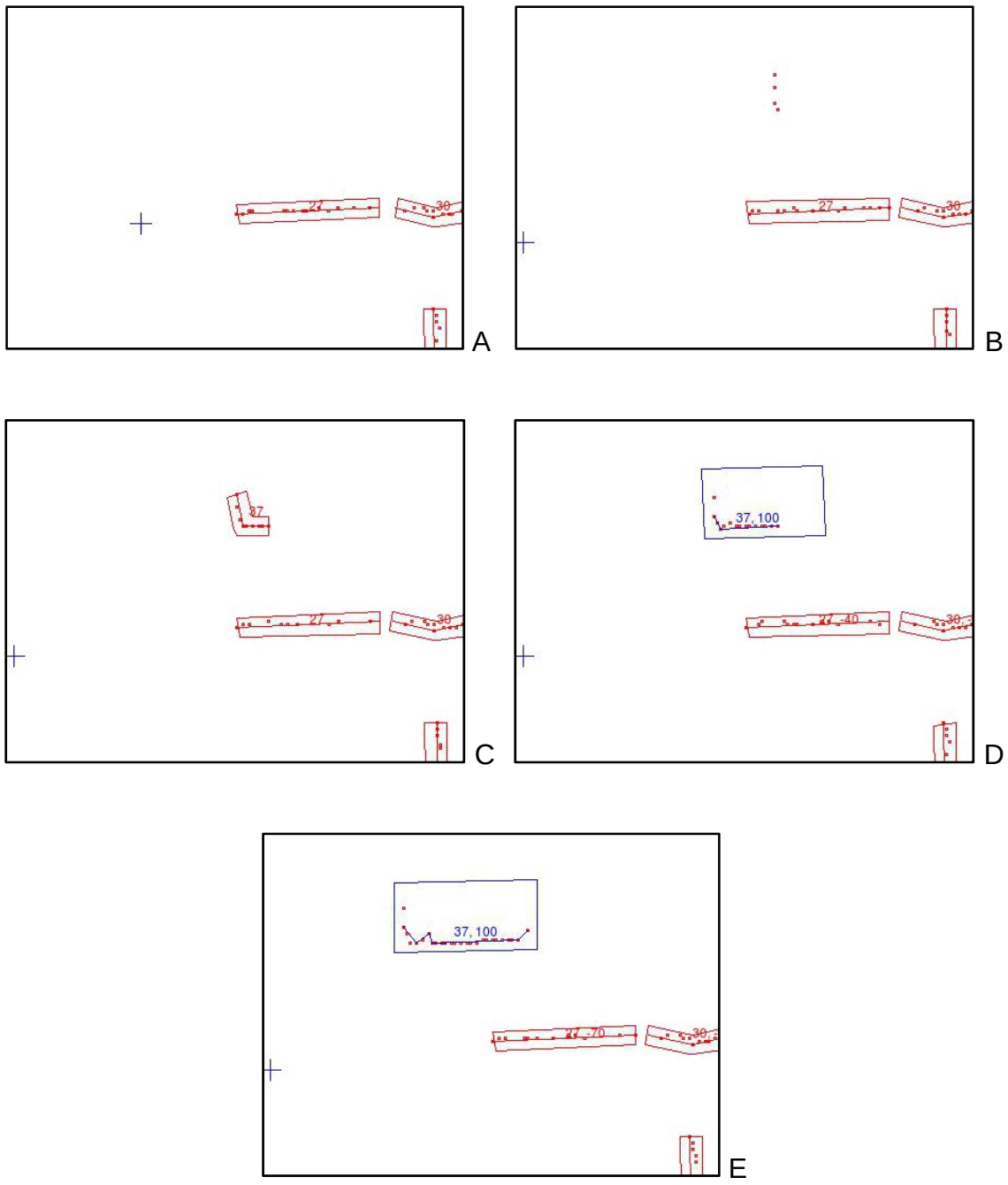



Figure 5-15. Occluded moving object is detected when it comes into view. A) Object is completely occluded. B) A few LADAR strike hot the object but it still cannot be detected by the detection system. C) The object is detected and initially assumed to be static. D) The object moving confidence passed the threshold and the object is converted to a moving object. E) The object is completely in view and the bounding box is updated.

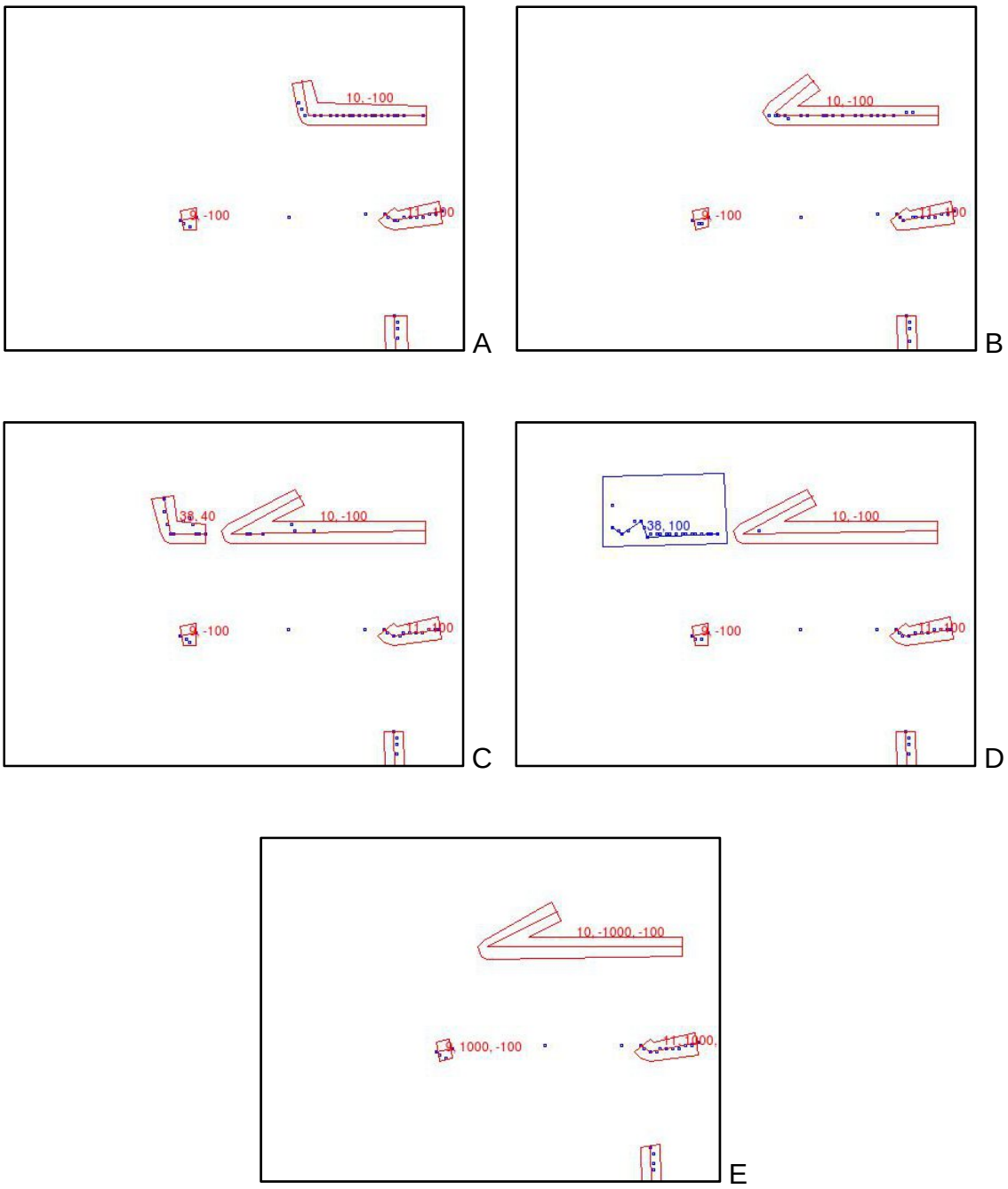



Figure 5-16. Static object becomes a moving object. A) The object is initially detected when it is stationary and is classified as a static object. B) The object starts to move but the moving confidence does not cause it to be classified as moving so the object is updated as a static object. C) Occlusion causes the object to be split into two. D) The new half of the object is classified as a moving object leaving old half. E) The old object existence confidence has been decreased to the minimum.

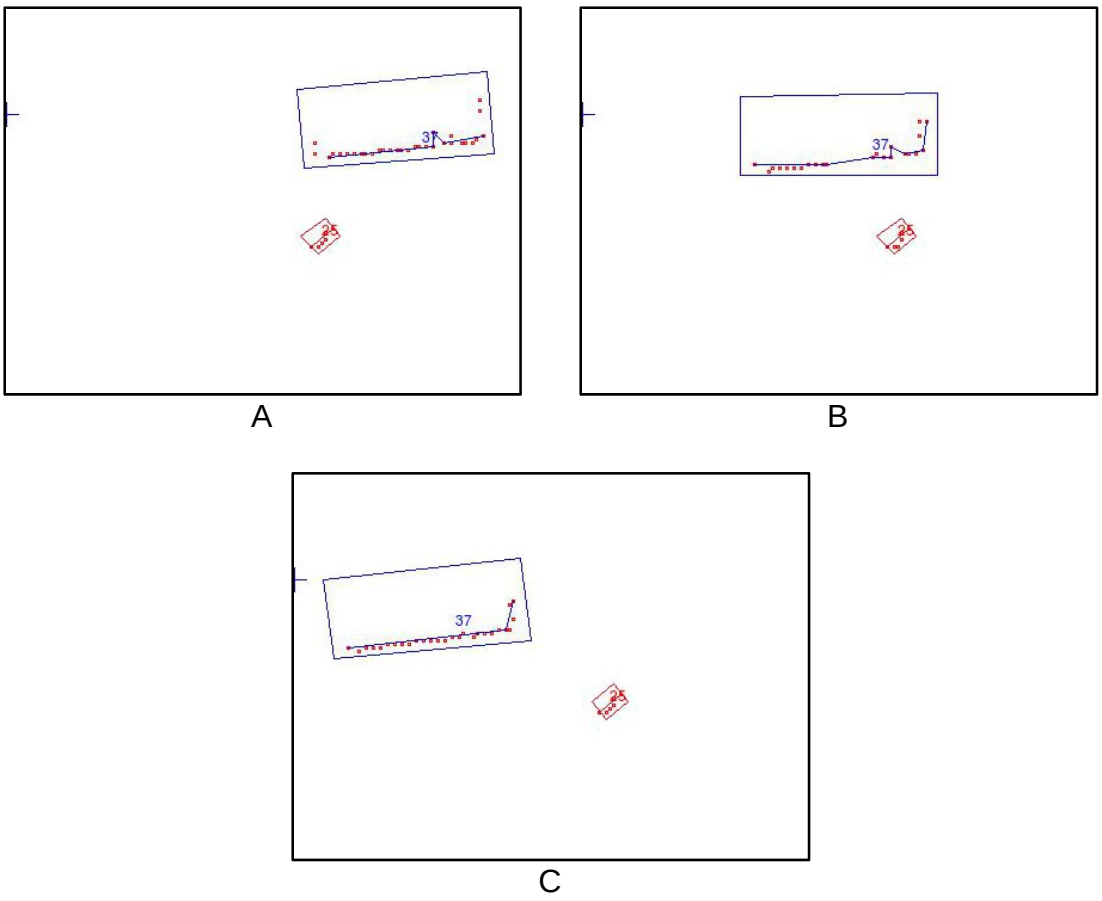



Figure 5-17. Moving object is partially occluded. A) The moving object starts to pass behind another object. B) The moving object is partially occluded but is correctly detected and tracked due to the use of the oriented bounding box. C). The object is no longer occluded and was never lost.

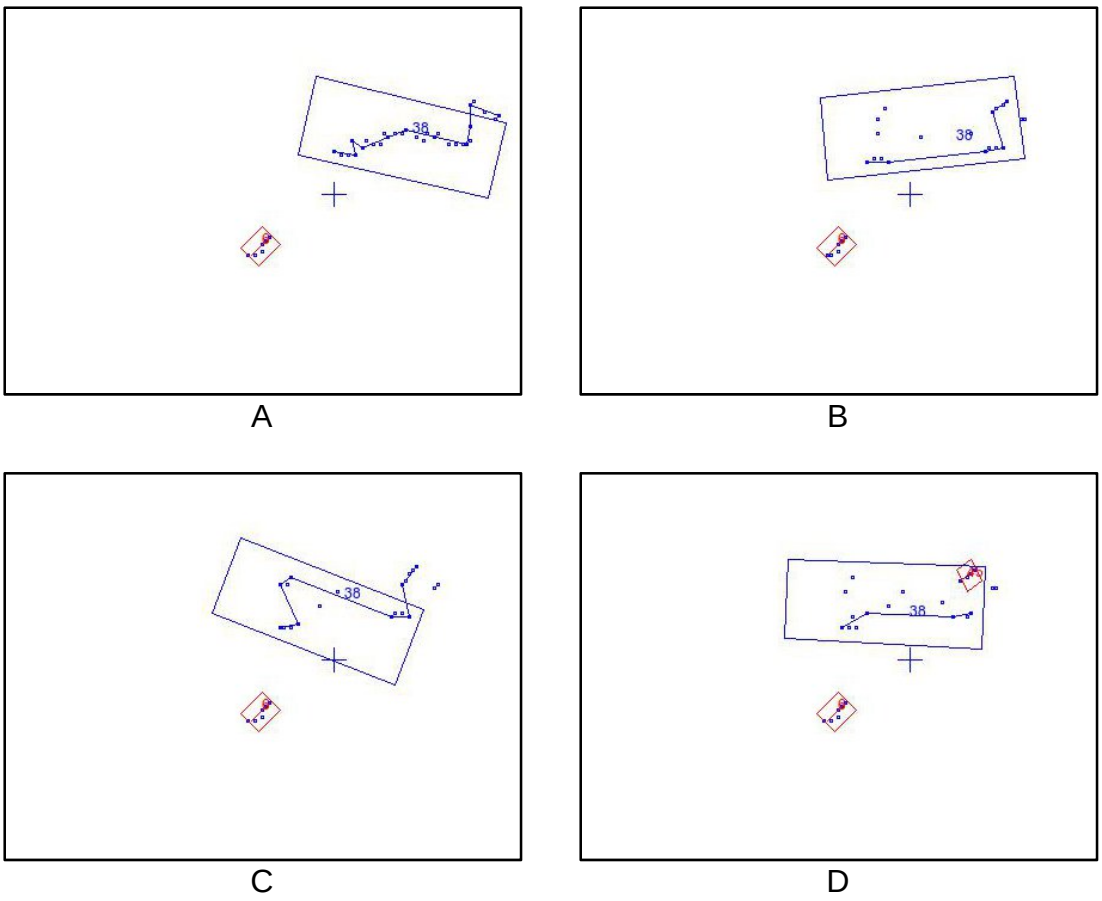



Figure 5-18. Object tracking degradation due sparse LADAR strikes. A) An unexpected object shape is detected due to the position of the sensor. B) The number of points striking the object starts to decrease. However, the object is still successfully tracked. C) The position of the bounding box changes drastically between scans. D) Static objects start to appear as the points are no longer continuous or closed enough together.

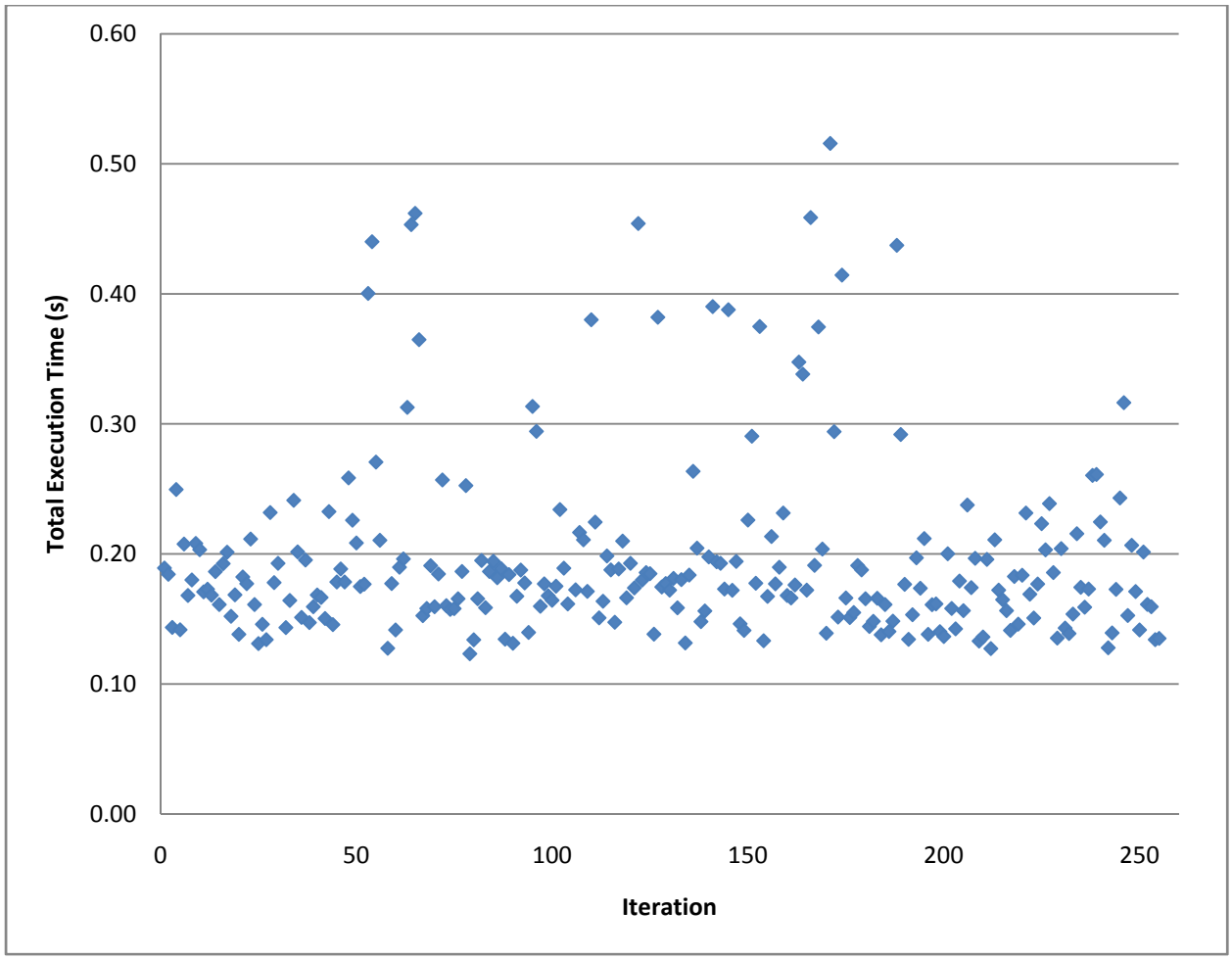



Figure 5-19. Total execution times for the system using the driver side LADAR with a static vehicle and dynamic environment with the presence of sensor noise, but without the use of position estimation, or access to the world model knowledge store.

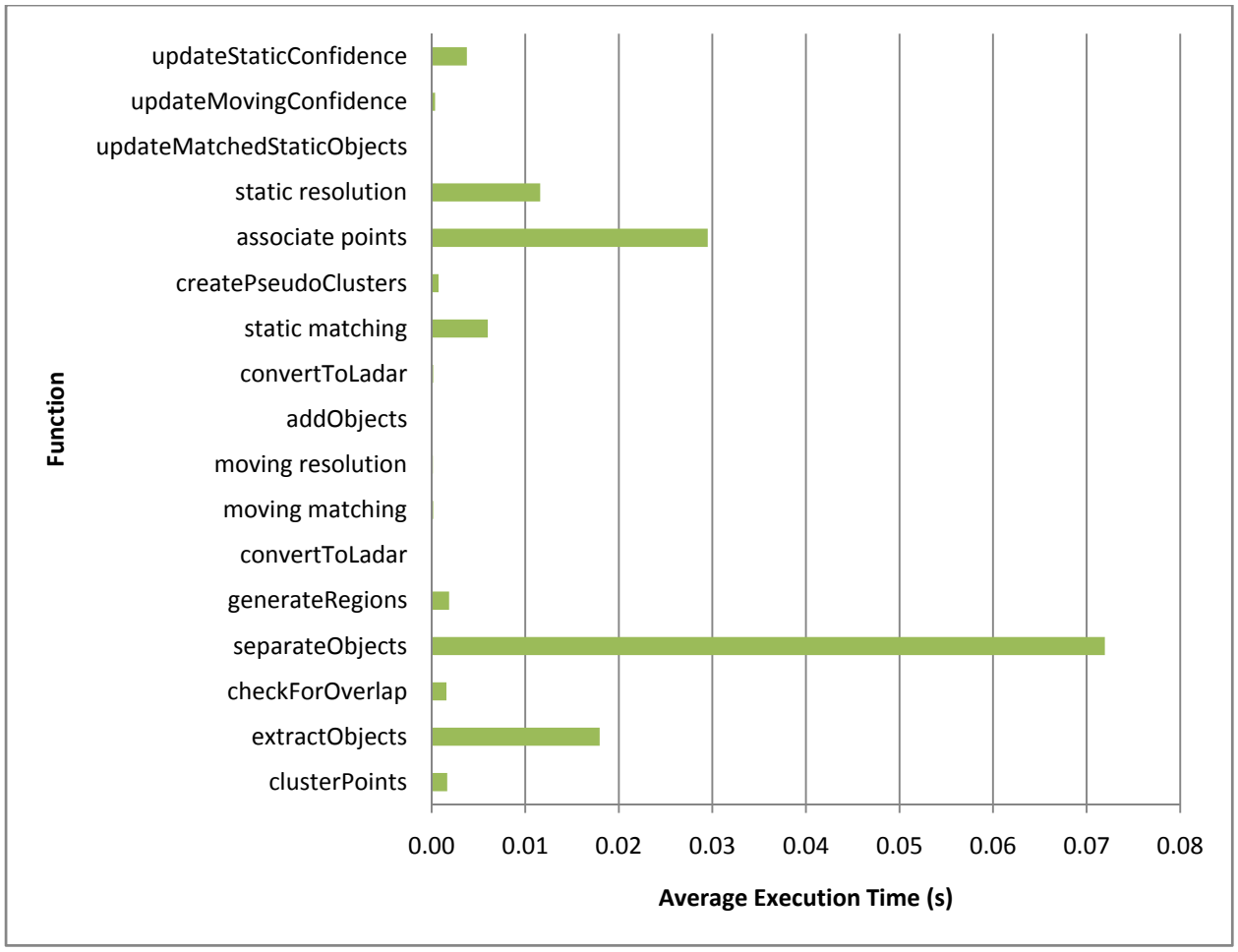



Figure 5-20. Average execution times for different functions using the driver side LADAR with a static vehicle and dynamic environment with the presence of sensor noise but without the use of position estimation, or access to the world model knowledge store.

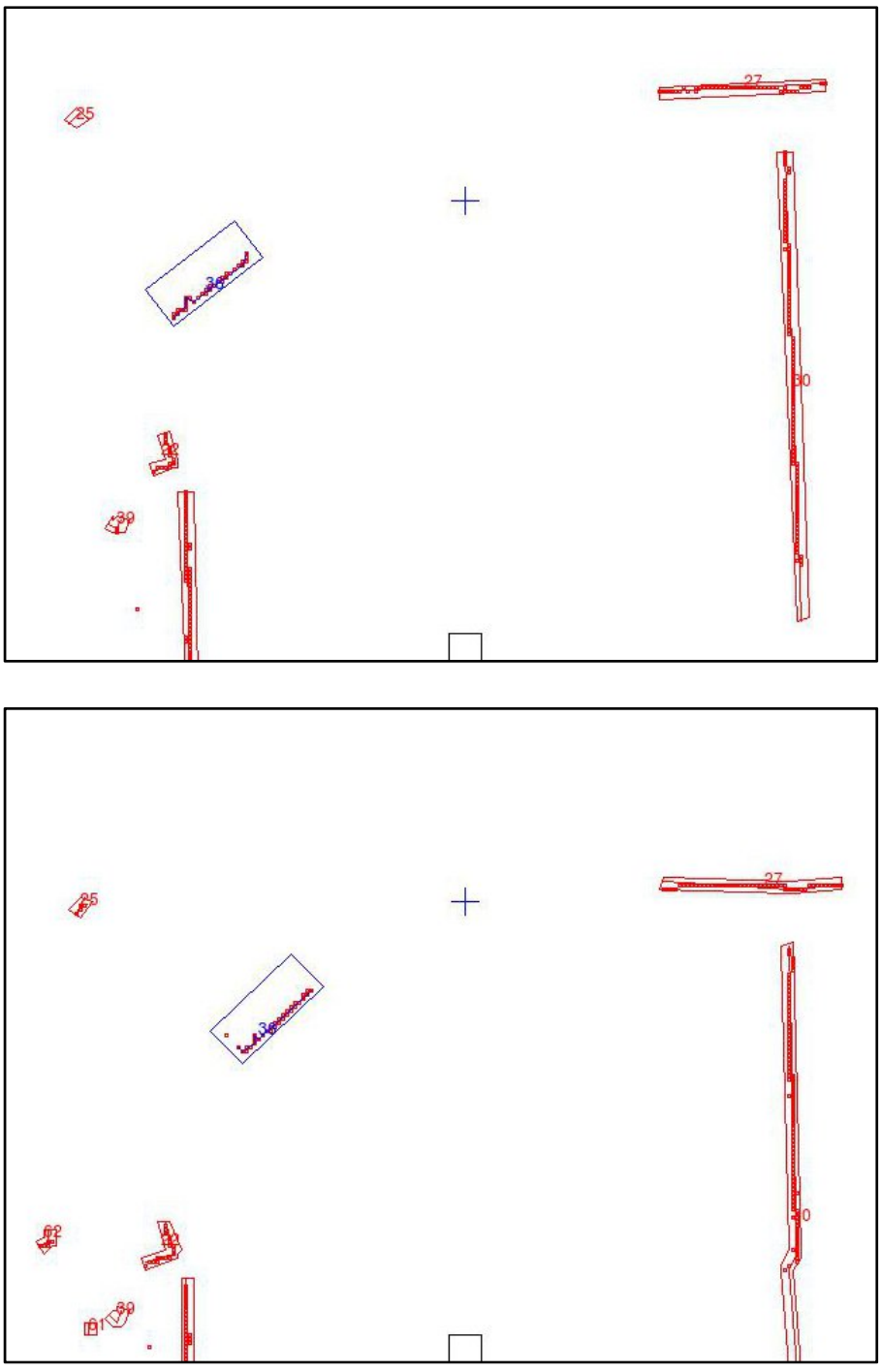
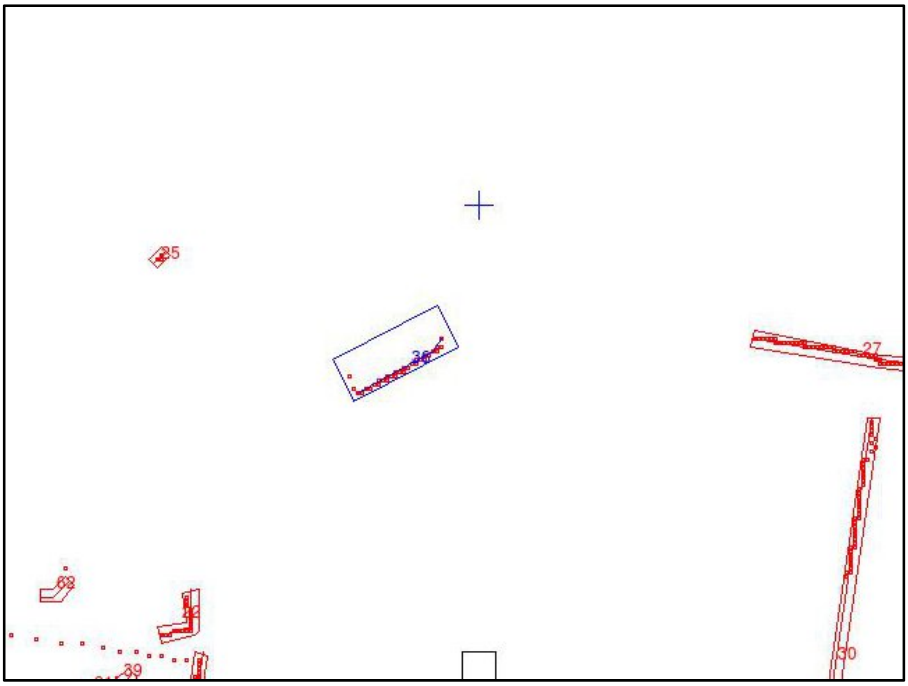
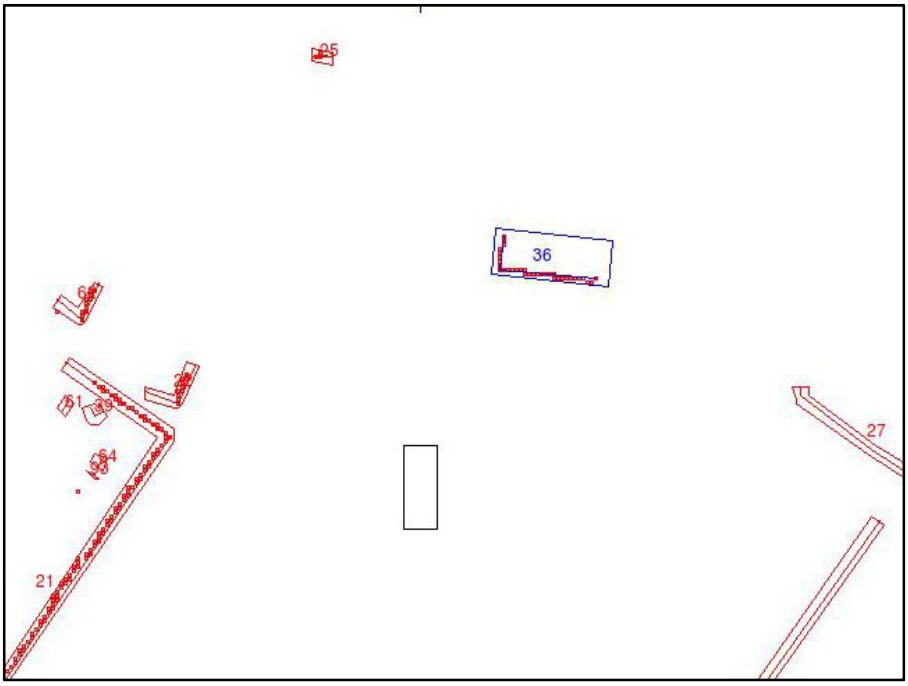



Figure 5-21. Moving object successfully tracked as the platform moves through the environment. A) The object is detected before the platform begins to move. B) The platform starts moving towards the moving object. C) The platform begins to turn and can still track the object. D) The moving object continues to be tracked as the static object representations are updated.




C




D

Figure 5-20. Continued.

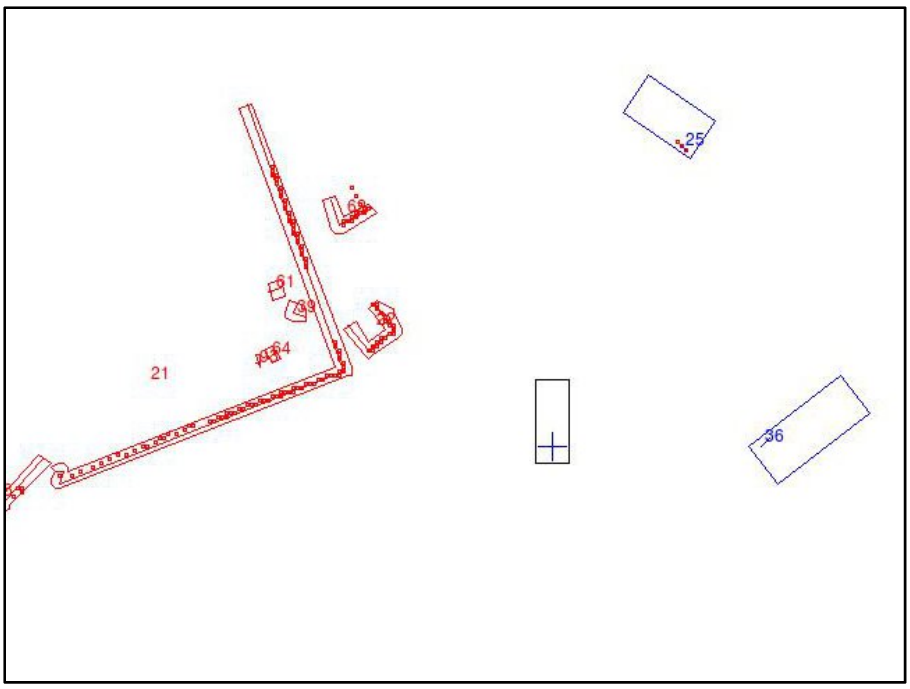



Figure 5-22. Object 25 incorrectly identified as a moving object due to error introduced through platform motion.

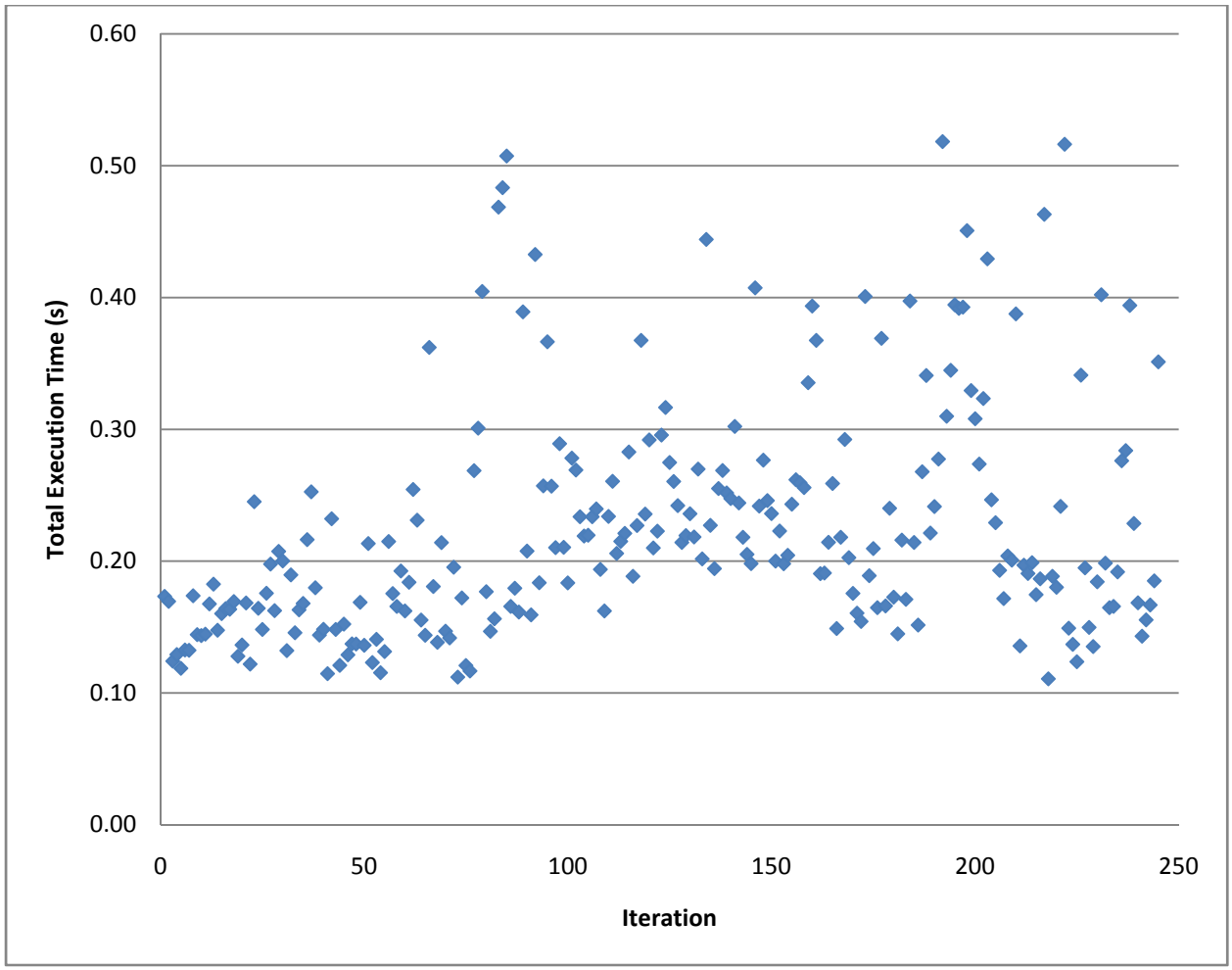



Figure 5-23. Total execution times for the system using the passenger side LADAR with a dynamic vehicle and dynamic environment with the presence of sensor noise, but without the use of position estimation, or access to the world model knowledge store.

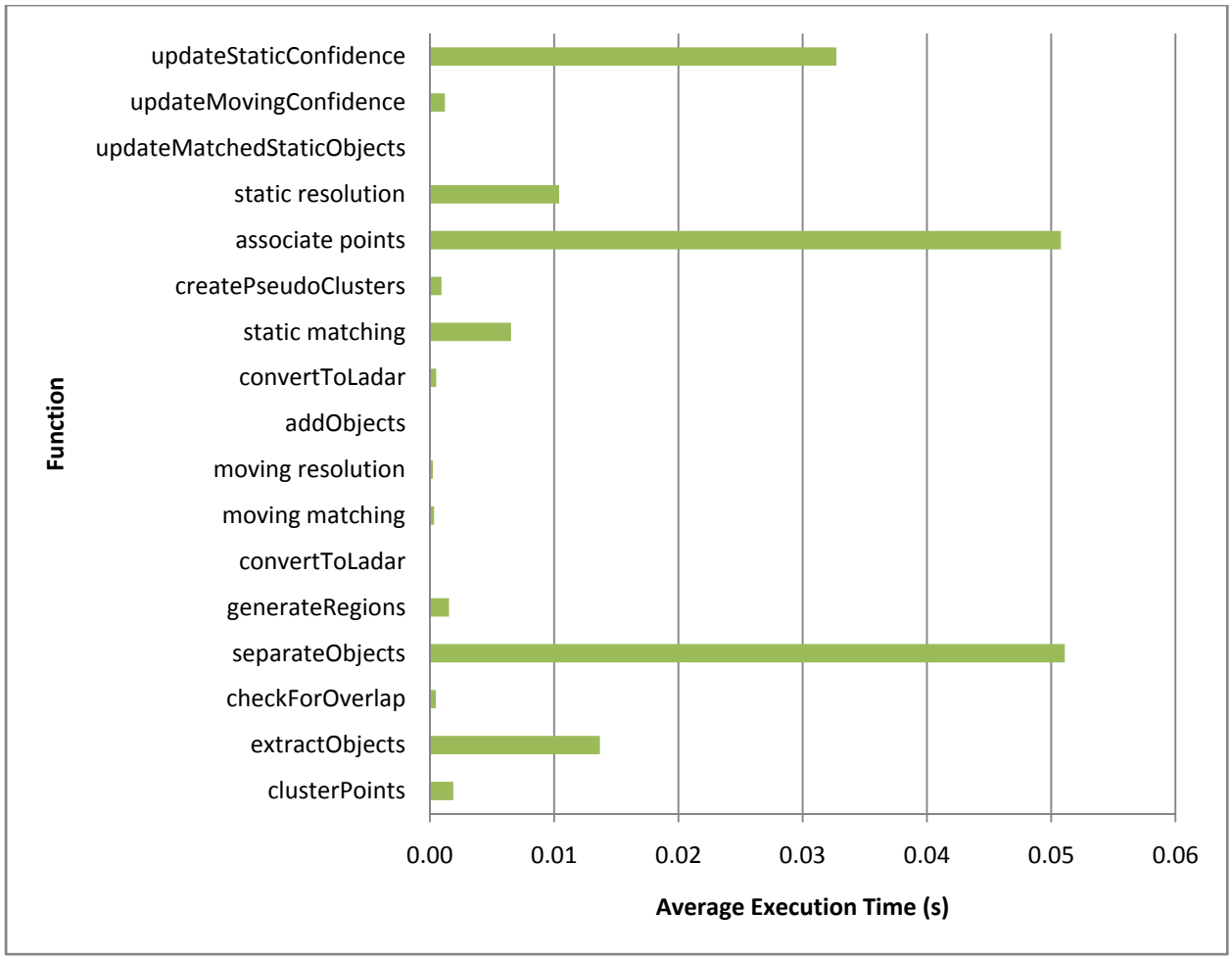



Figure 5-24. Average execution times for different functions using the passenger side LADAR with a dynamic vehicle and dynamic environment with the presence of sensor noise but without the use of position estimation, or access to the world model knowledge store.

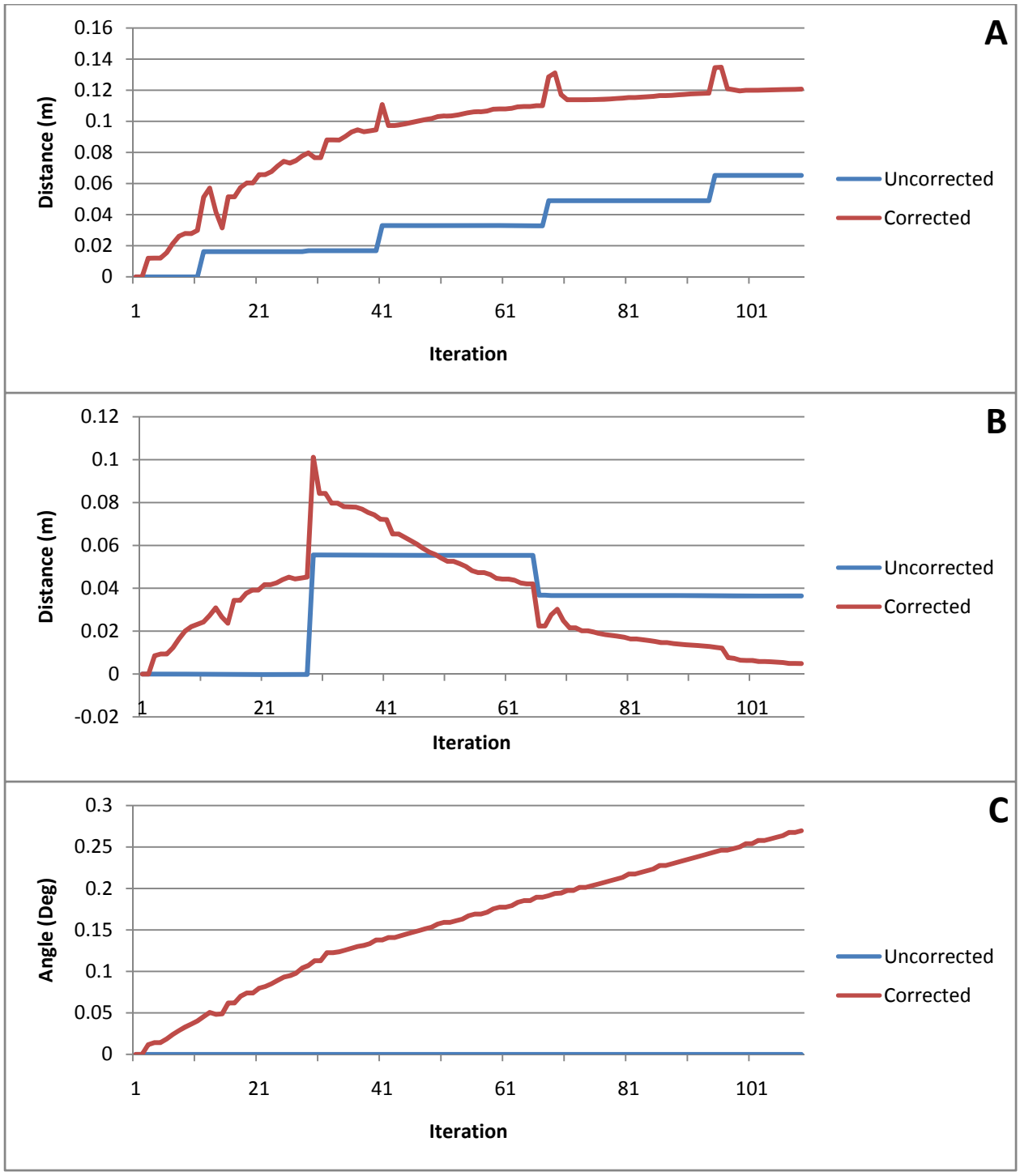



Figure 5-25. Distance from the origin of the corrected and uncorrected position estimates when running with fixed LADAR data on a static platform in a static environment. A) Change in the UTM X position. B) Change in the UTM Y position. C) Change in Yaw.

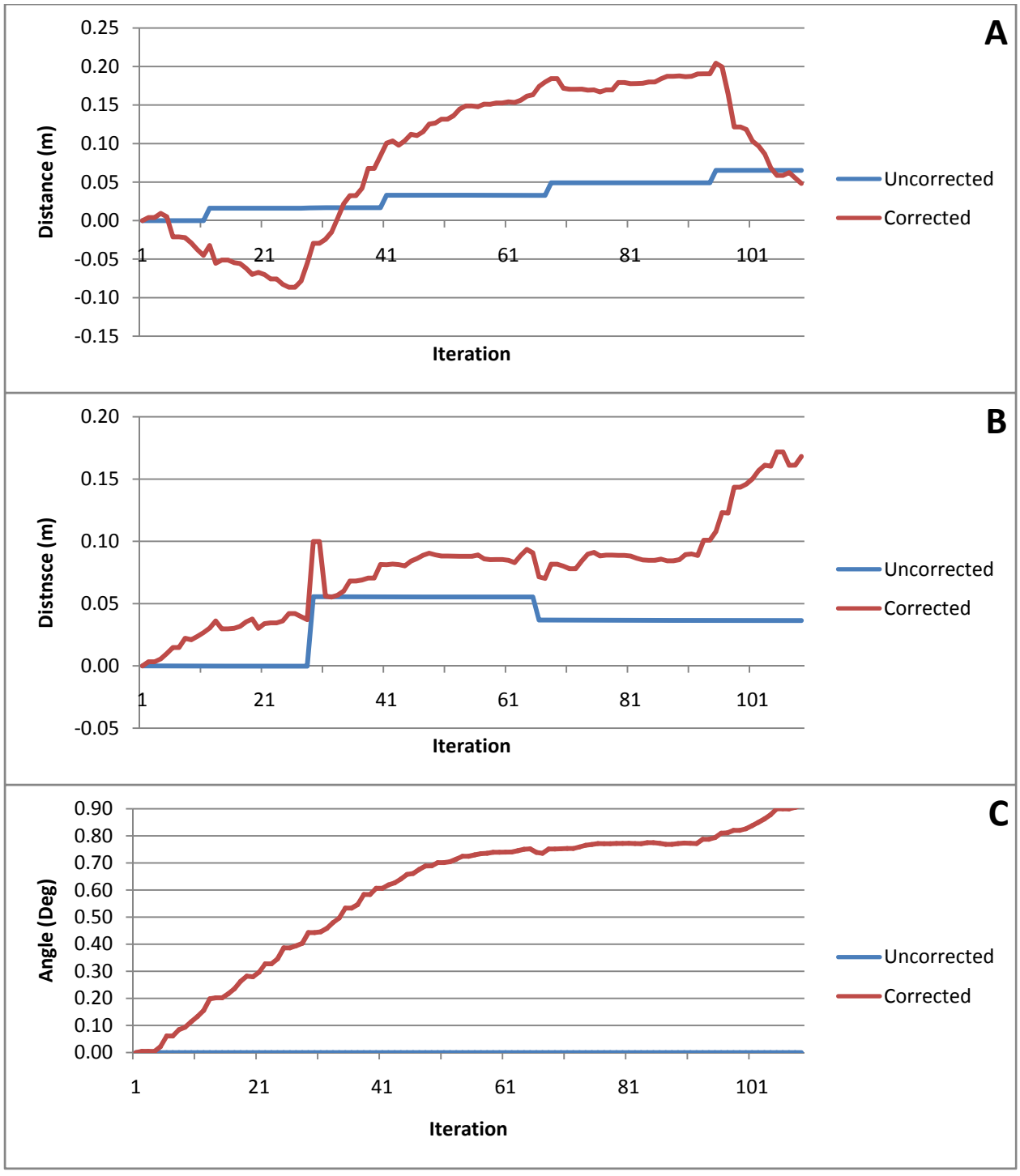



Figure 5-26. Distance from the origin of the corrected and uncorrected position estimates when running with real LADAR data on a static platform in a static environment. A) Change in the UTM X position. B) Change in the UTM Y position. C) Change in Yaw.

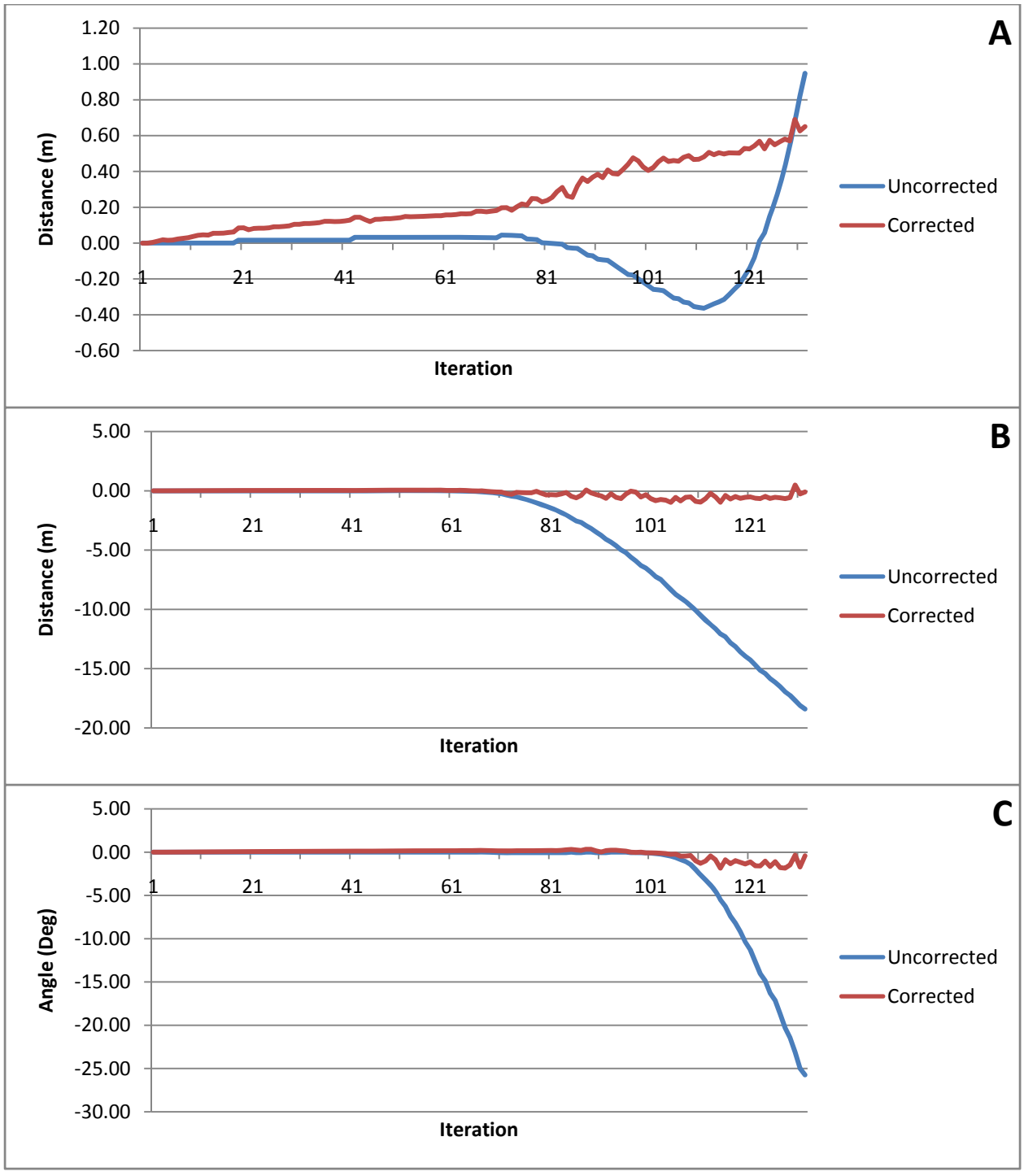



Figure 5-27. Distance from the origin of the corrected and uncorrected position estimates when running with fixed LADAR data on a dynamic platform in a static environment. A) Change in the UTM X position. B) Change in the UTM Y position. C) Change in Yaw.

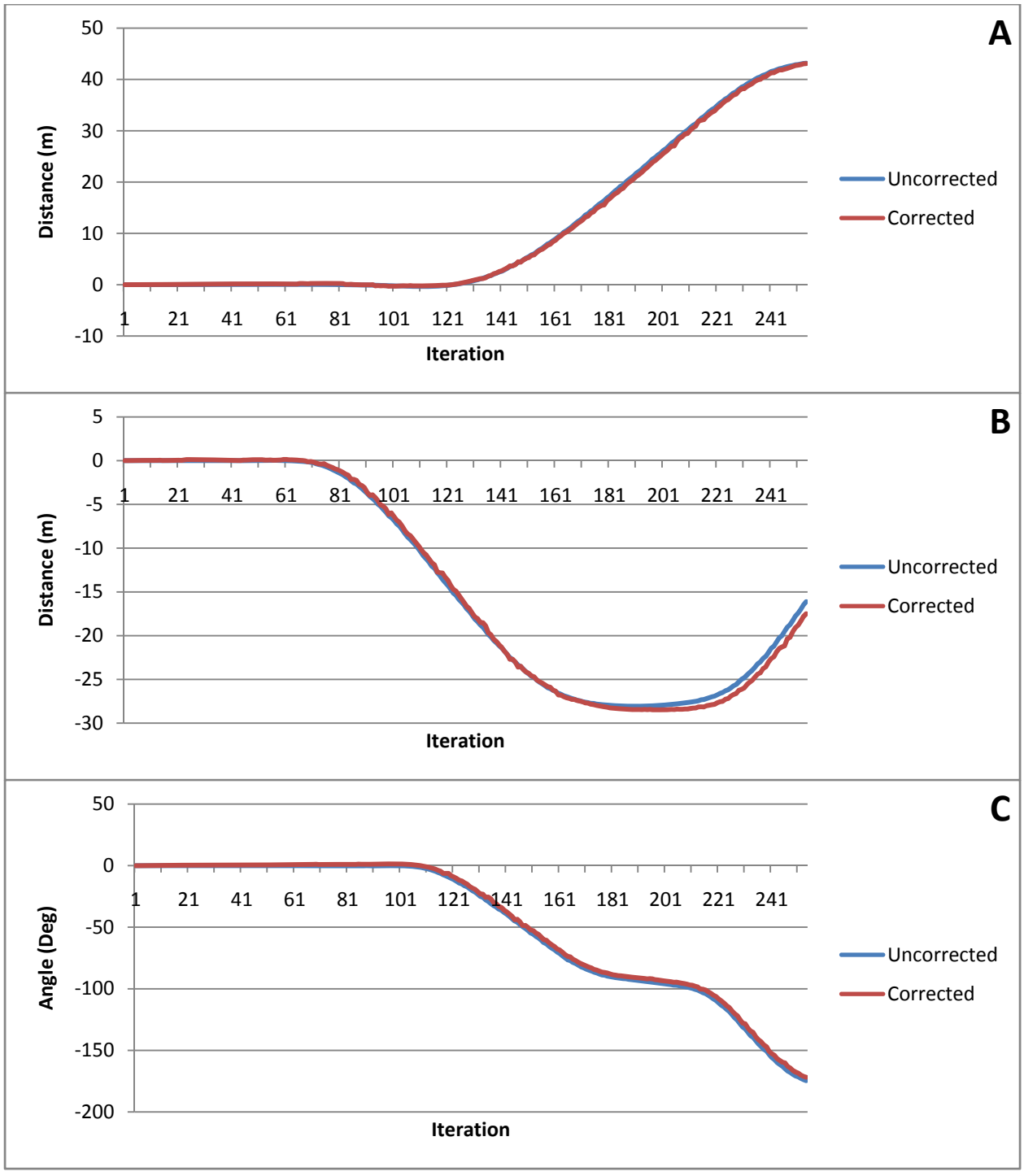



Figure 5-28. Distance from the origin of the corrected and uncorrected position estimates when running with real LADAR data on a dynamic platform in a static environment. A) Change in the UTM X position. B) Change in the UTM Y position. C) Change in Yaw.

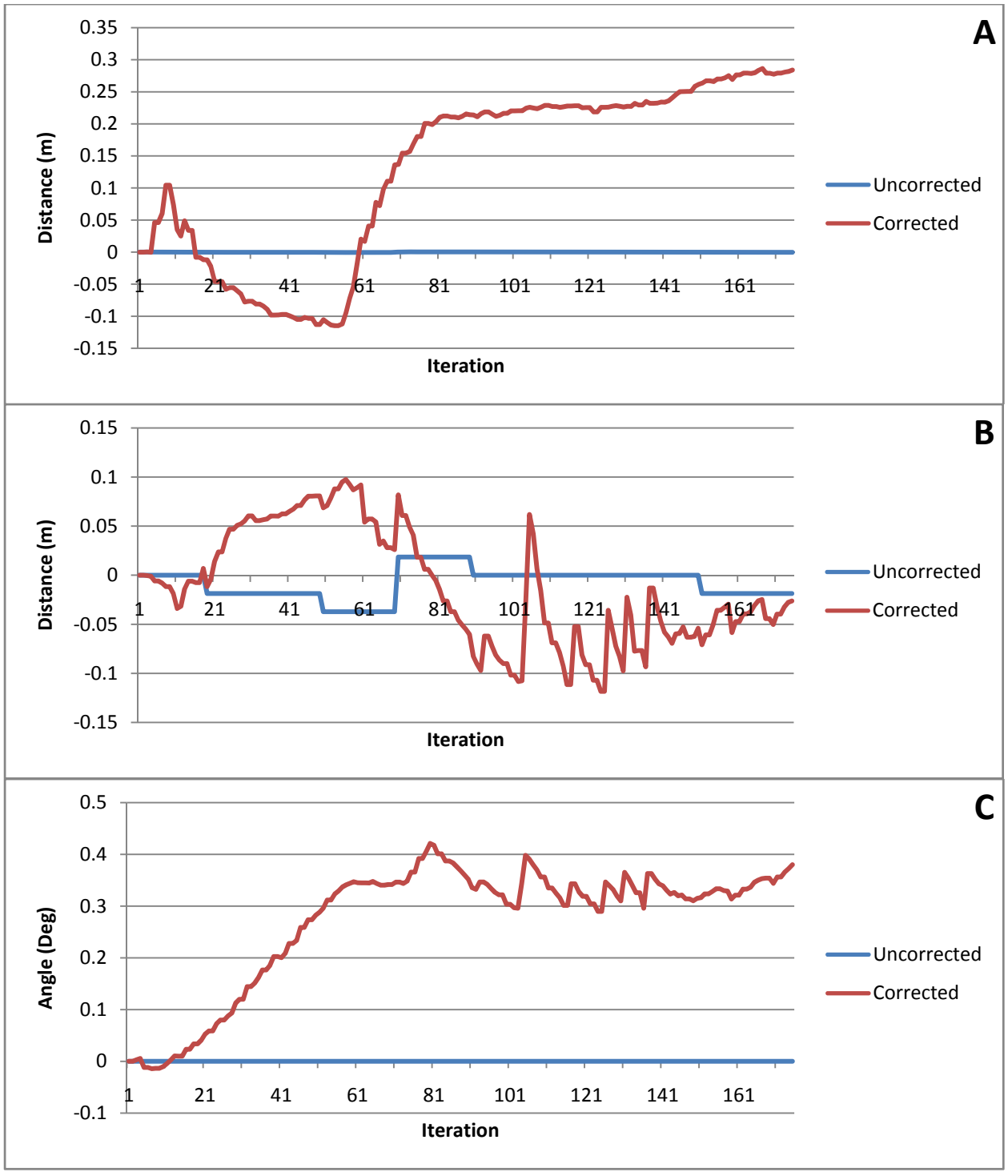
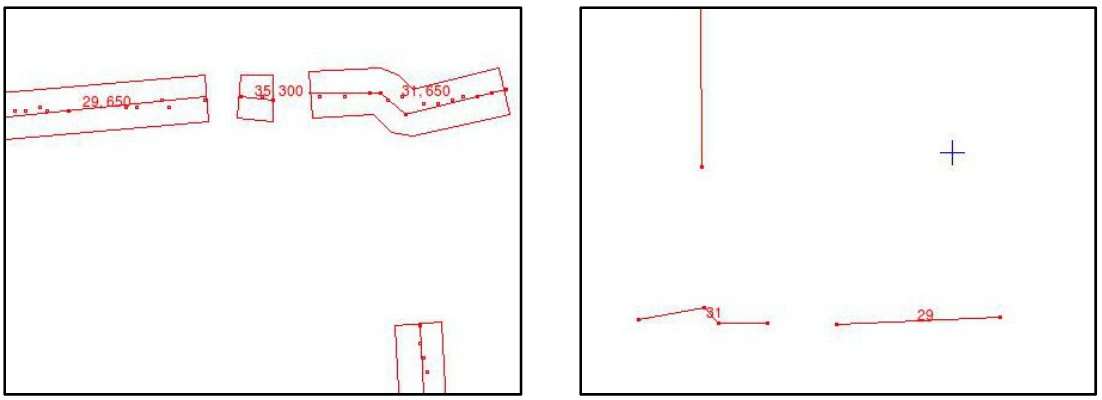
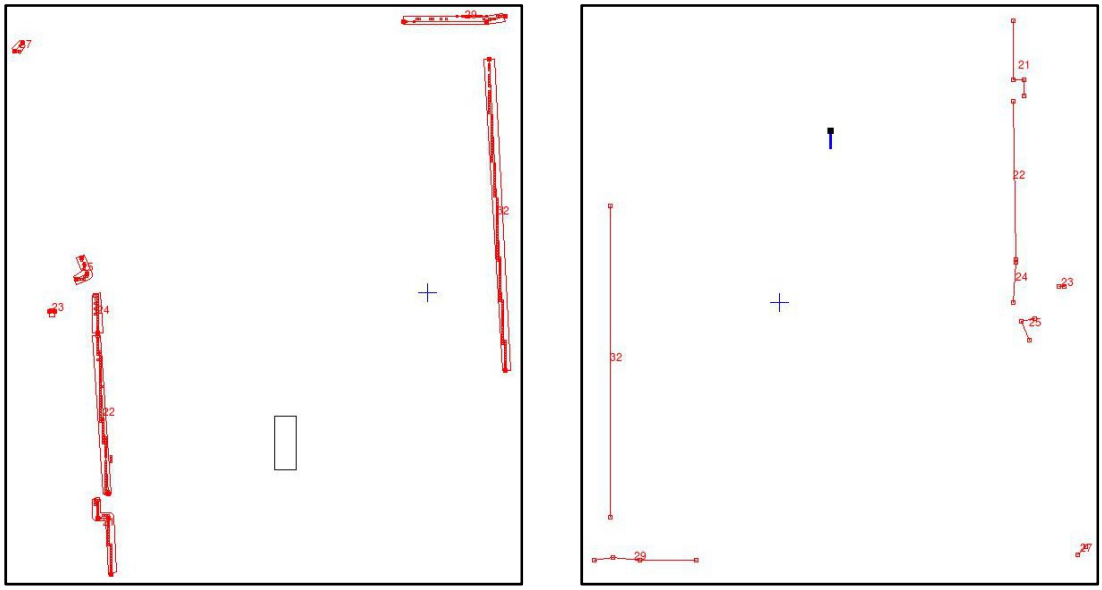



Figure 5-29. Distance from the origin of the corrected and uncorrected position estimates when running with real LADAR data on a static platform in a dynamic environment. A) Change in the UTM X position. B) Change in the UTM Y position. C) Change in Yaw.




A B

Figure 5-30. Objects are added to the WMKS. A) The existence confidence of objects 29 and 31 has passed the threshold and should be added to the WMKS. B) Objects 29 and 31 have been successfully added to the WMKS.




A B

Figure 5-31. Objects are updated over time. A) Object 29 was updated by merging two objects. B) Object 29 is updated in the WMKS.

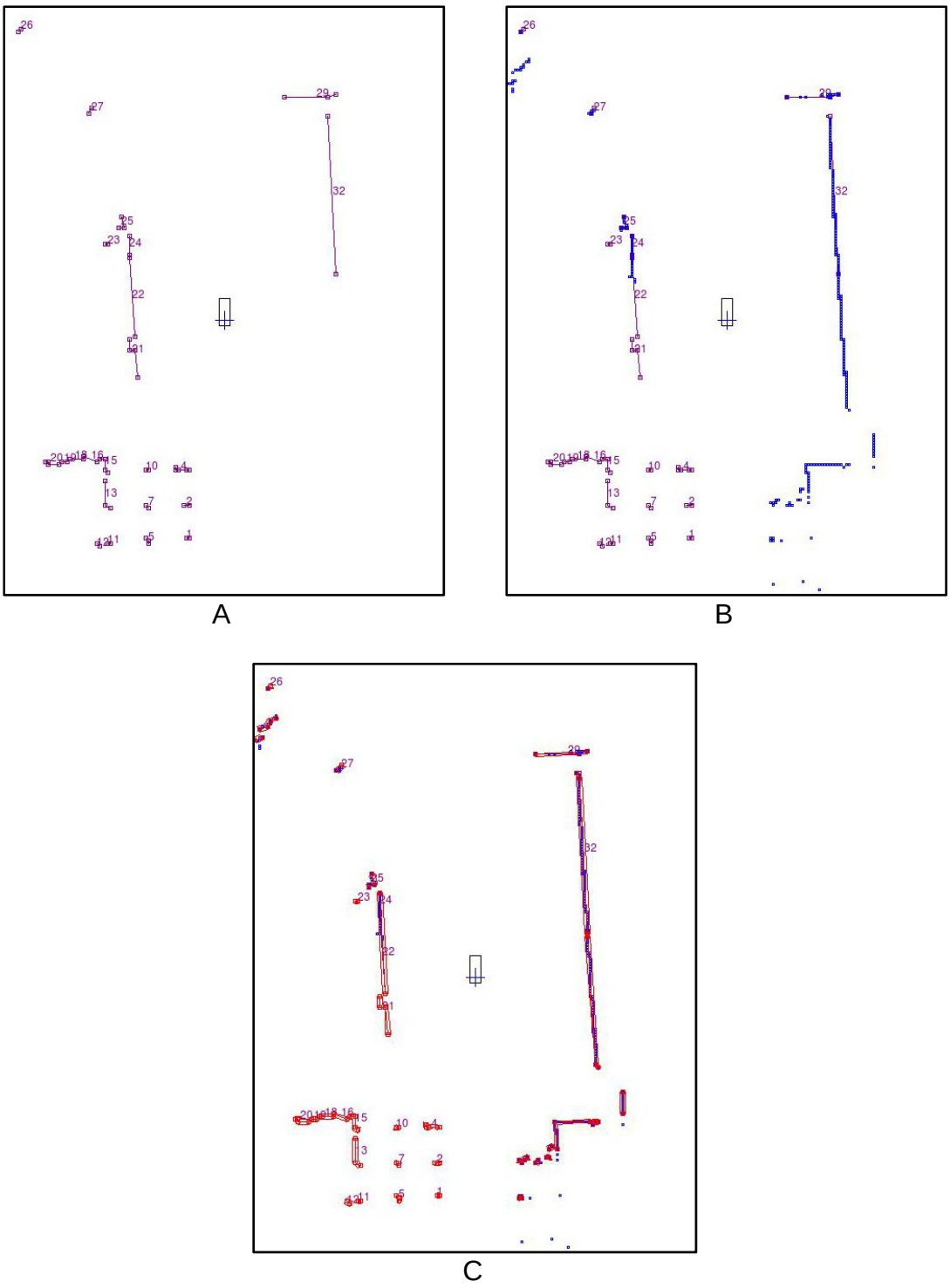



Figure 5-32. Stored objects are retrieved from the WMKS and updated using the new LADAR data. A) The objects in the WMKS are successfully retrieved. B) The LADAR points only correspond to some of the retrieved objects. C) The retrieved objects are updated using the current LADAR scan.

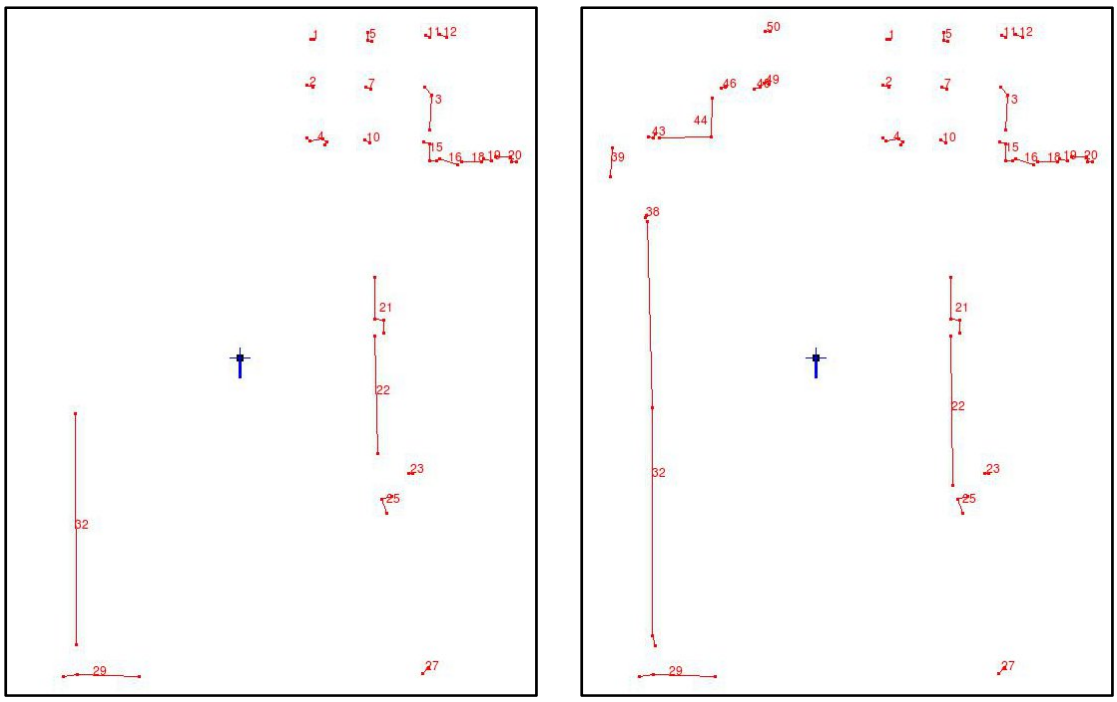



Figure 5-33. The WMKS is updated. A) The previously stored objects. B) Previously stored objects are updated (objects 32 and 22) and newly detected objects are added.

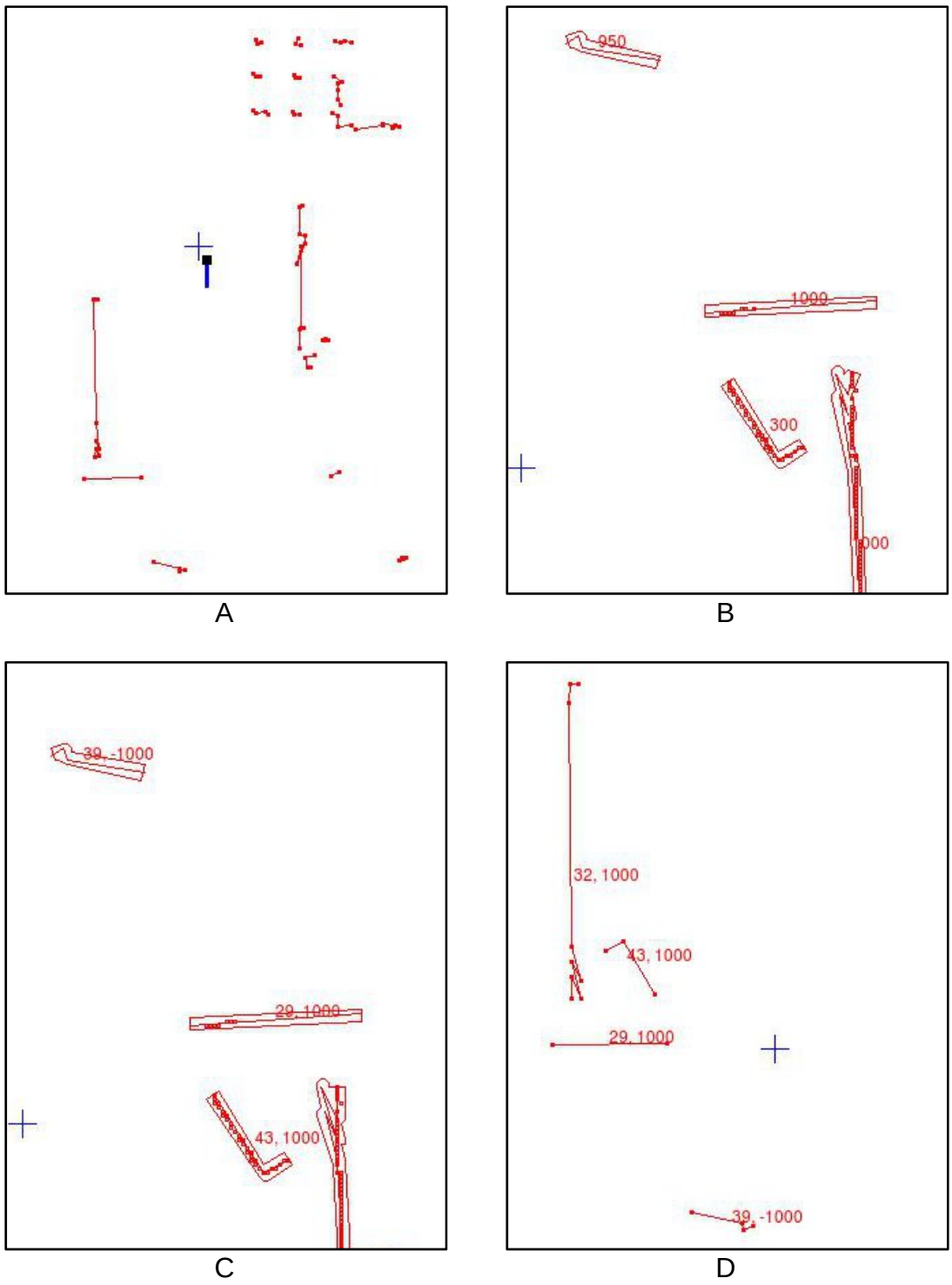
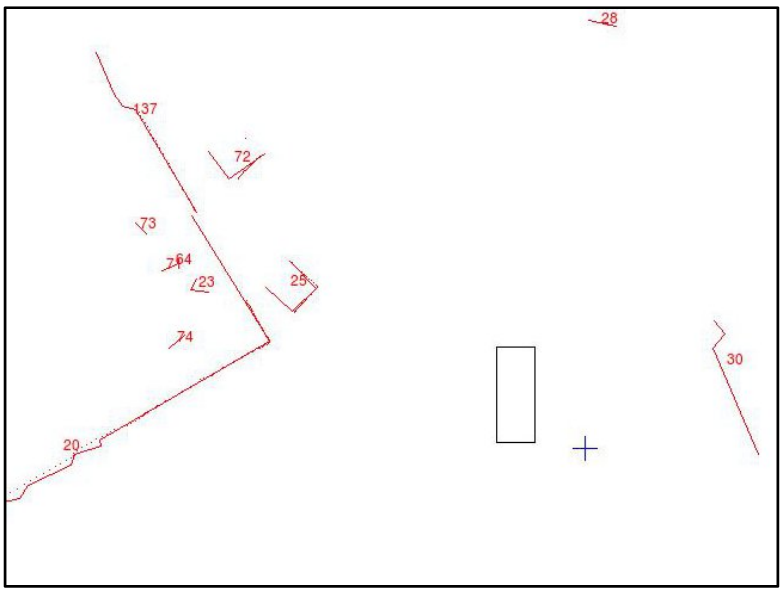
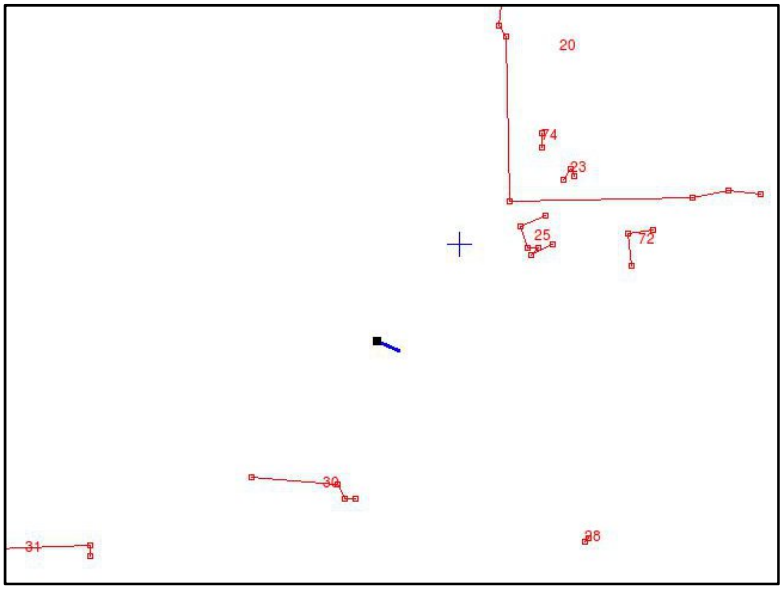



Figure 5-34. Object is detected as missing and new object is detected. A) Objects stored in WMKS. B) Previously stored object is not present and new object is detected. C) Confidence of missing object (30) goes to -1000 and new object (43) has a confidence of 1000. D) WMKS confidence is updated.

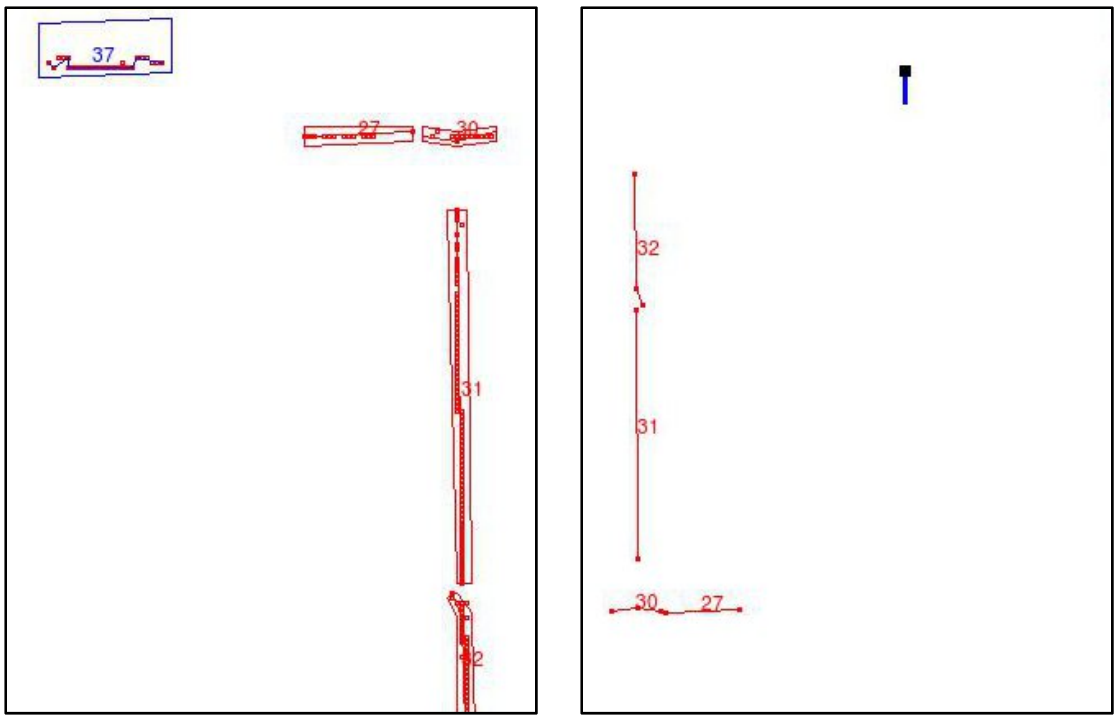



A




B

Figure 5-35. Reconstructed objects are successfully stored in the WMKS. A) The objects in the SLAM+DATMO system. B) The objects in the WMKS.




A B

Figure 5-36. Moving Objects are not added to the WMKS. A) Moving object 37 is detected by the SLAM+DATMO system. B) Object 37 is not added to the WMKS.

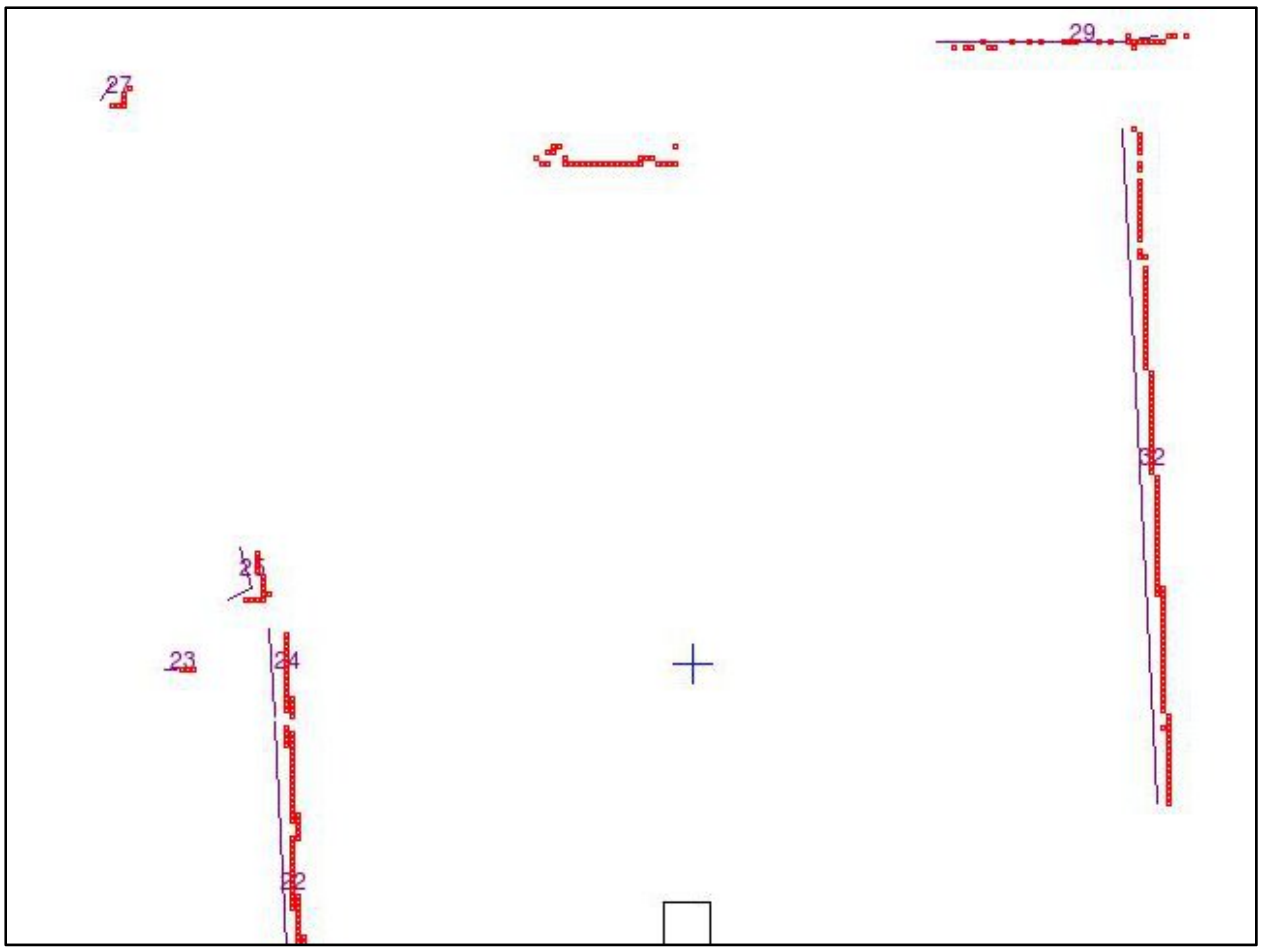



Figure 5-37. Objects stored in WMKS are not aligned the current LADAR scan.

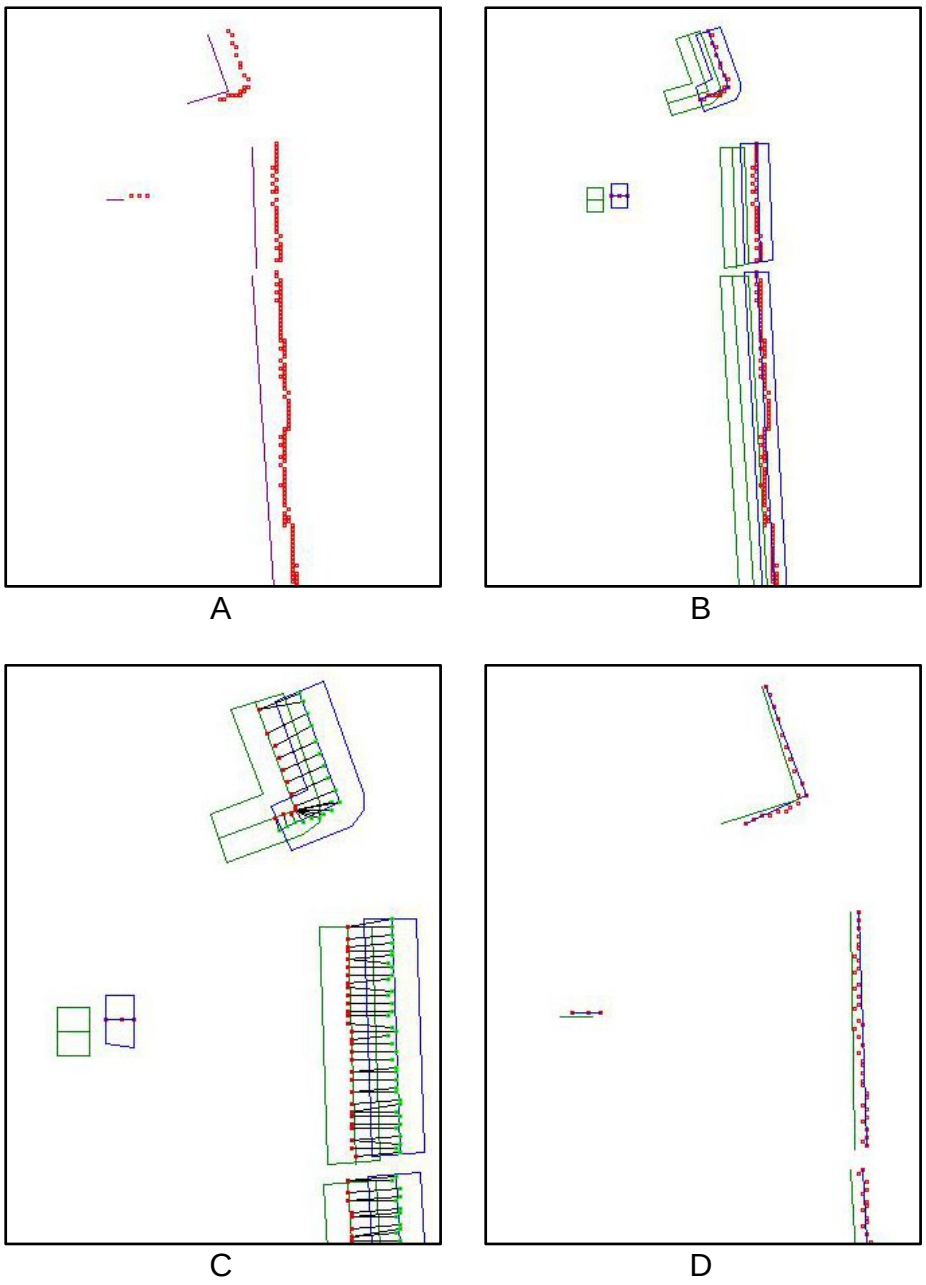



Figure 5-38. Objects are matched and the current position is updated. A) The WMKS objects and the current LADAR scan are not aligned. B) The objects extracted using the current scan match some of the stored WMKS objects. C) The object points are associated in order to perform the position correction. D) The vehicle position is updated and causes the objects to become closer to being aligned.

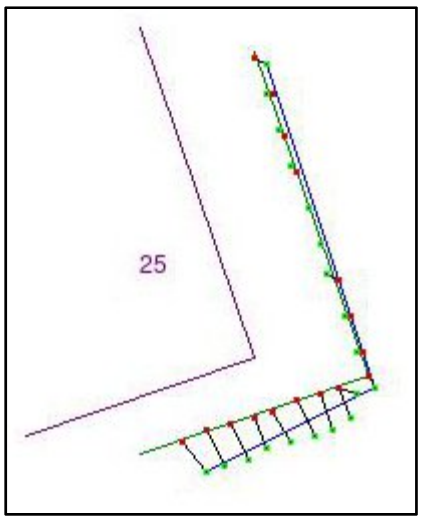



Figure 5-39. WMKS object (purple) versus corrected stored objects (green with red points) versus extracted objects (blue with green points)

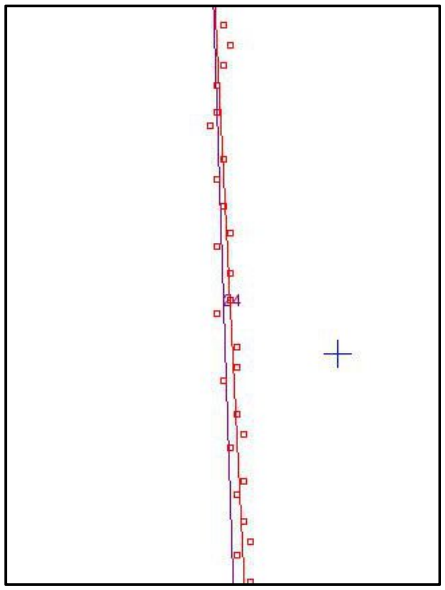


Figure 5-40. The correct position causes the WMKS objects (purple) to become aligned with the sensed objects (red).

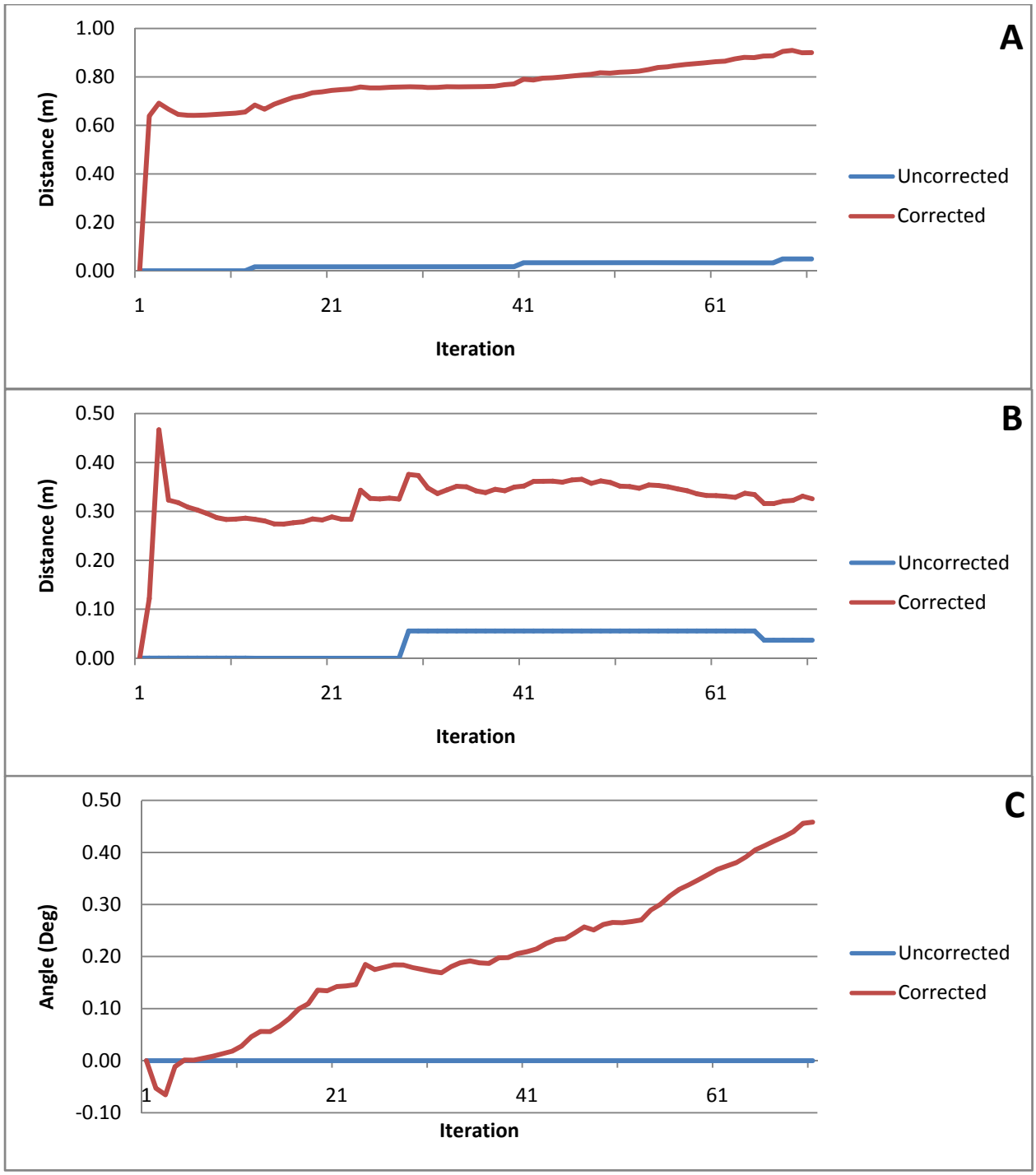



Figure 5-41. Distance from the origin of the corrected and uncorrected position estimates when running with real LADAR data on a static platform in a static environment with a difference between the retrieved WMKS objects and the sensed objects. A) Change in the UTM X position. B) Change in the UTM Y position. C) Change in Yaw.

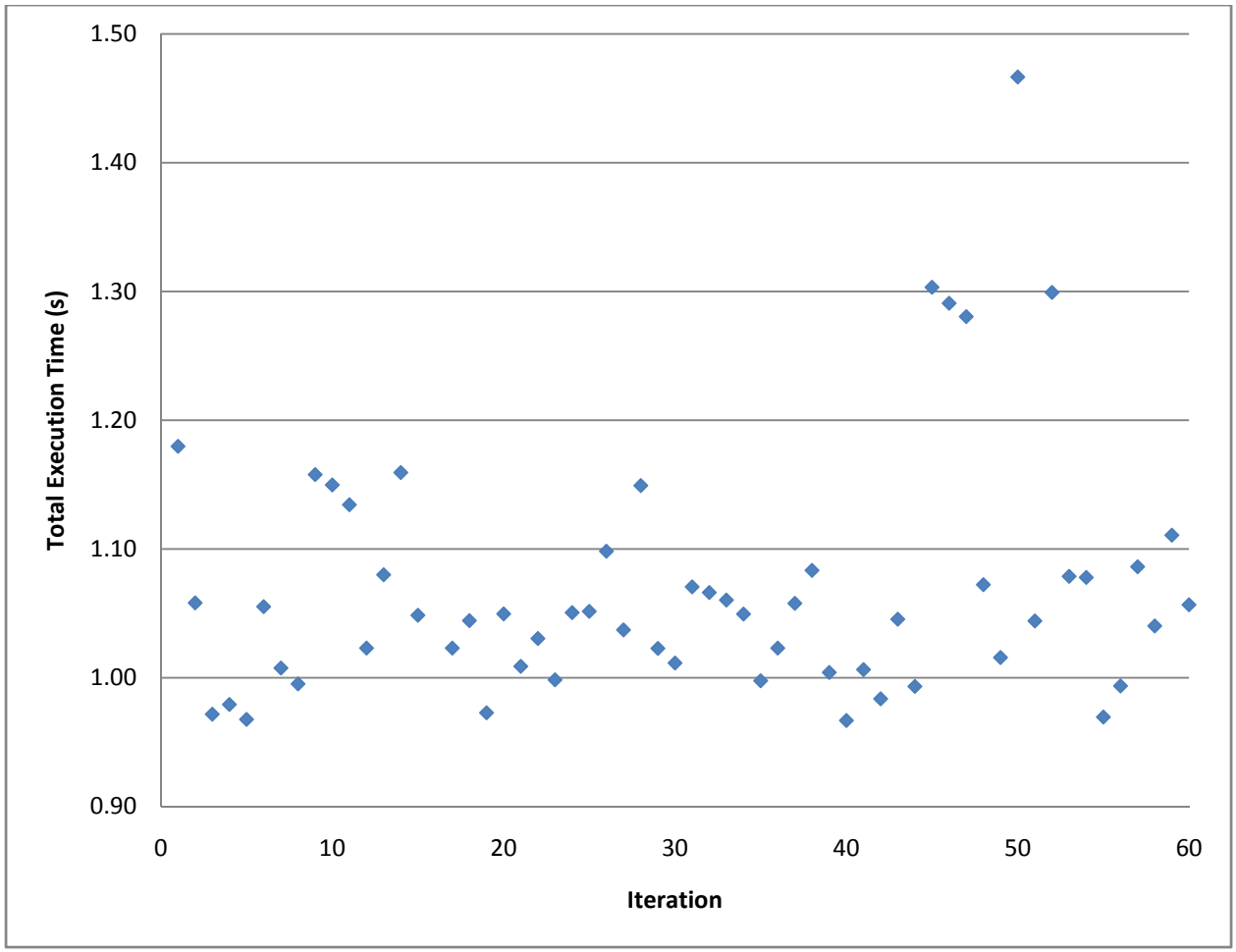



Figure 5-42. Total execution times for the system using the driver side LADAR with a static vehicle and static environment with the presence of sensor noise, the use of position estimation, and access to the world model knowledge store.

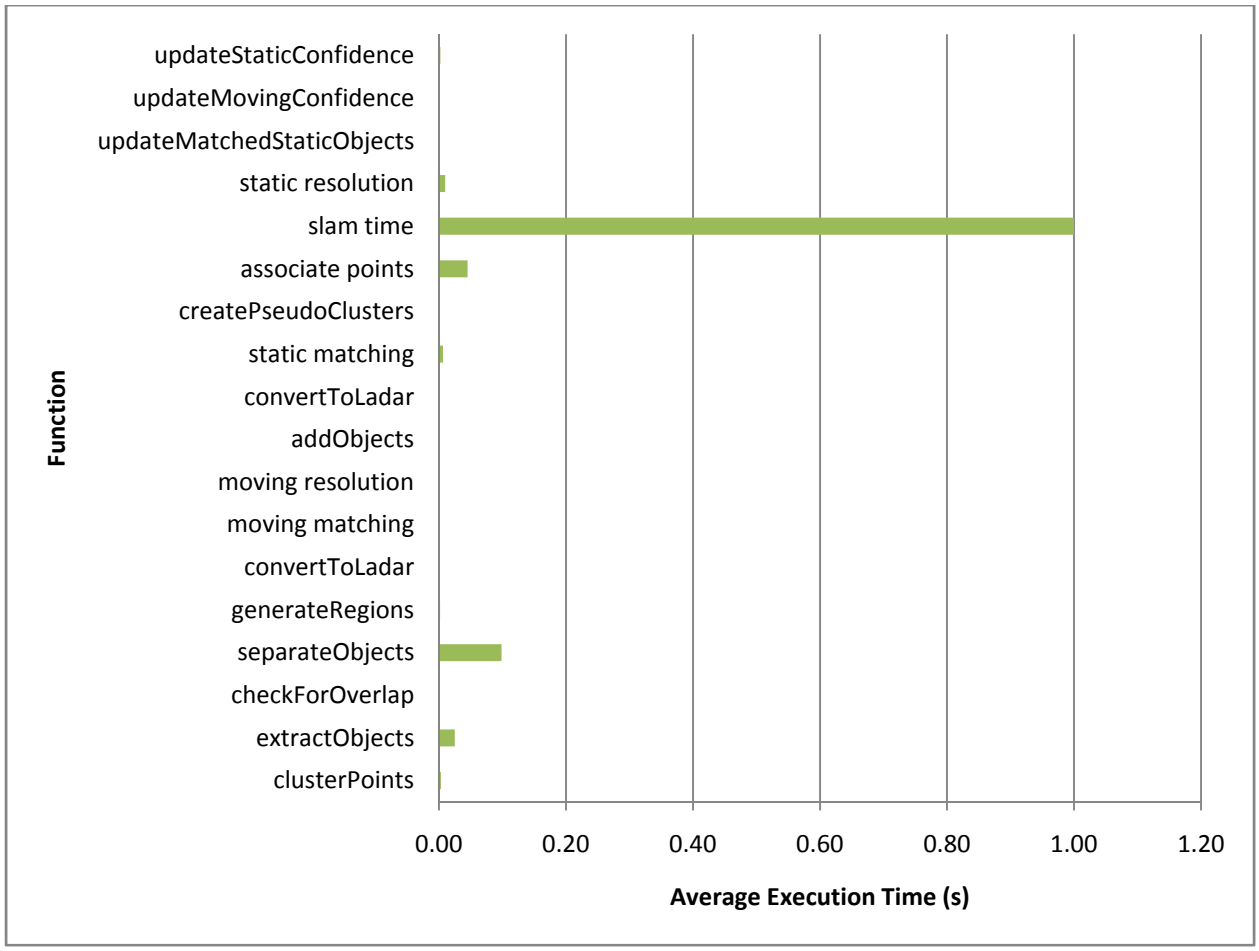



Figure 5-43. Average execution times for different functions using the driver side LADAR with a static vehicle and static environment with the presence of sensor noise, the use of position estimation, and access to the world model knowledge store.

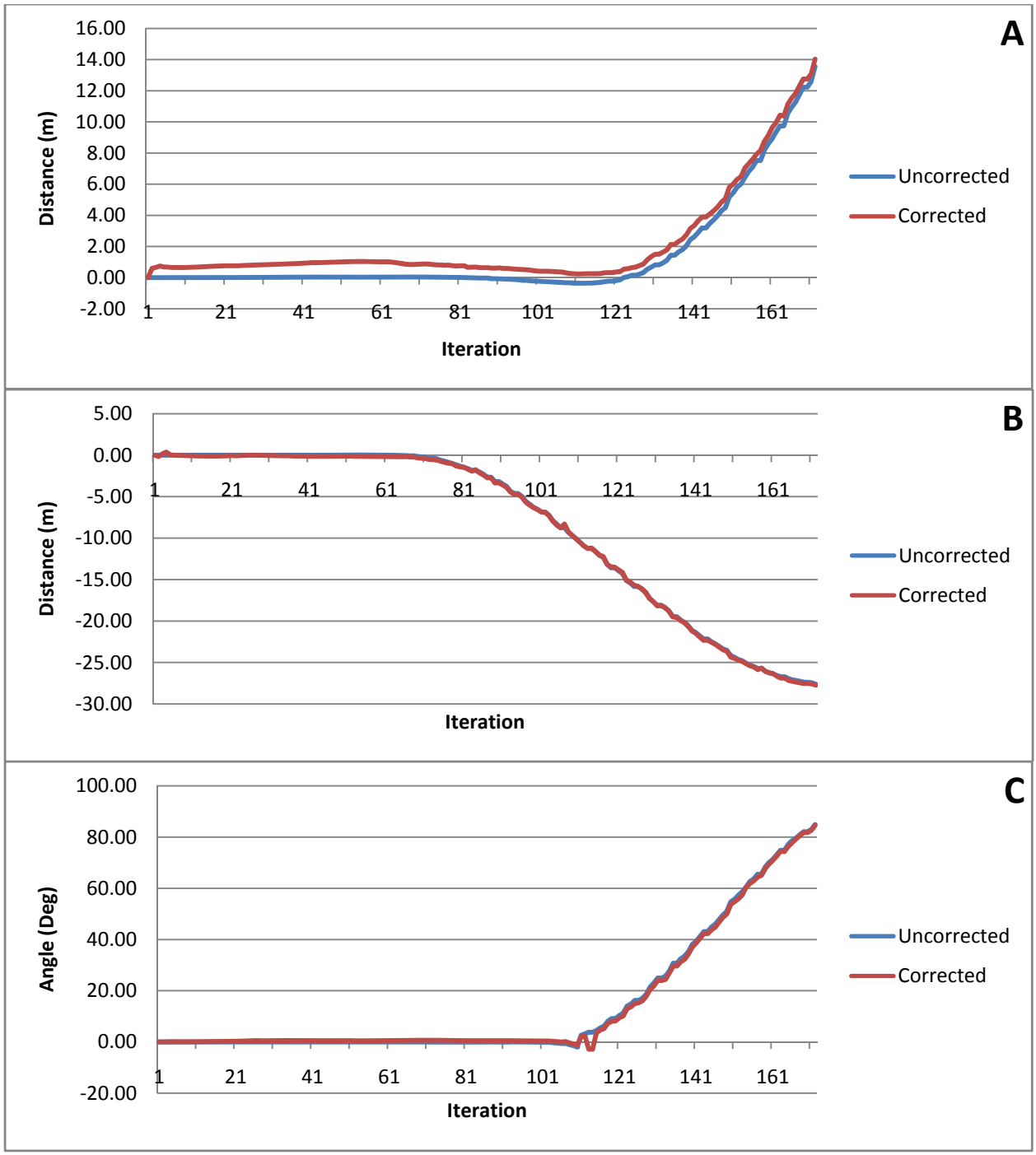



Figure 5-44. Distance from the origin of the corrected and uncorrected position estimates when running with real LADAR data on a dynamic platform in a static environment with a difference between the retrieved WMKS objects and the sensed objects. A) Change in the UTM X position. B) Change in the UTM Y position. C) Change in Yaw.

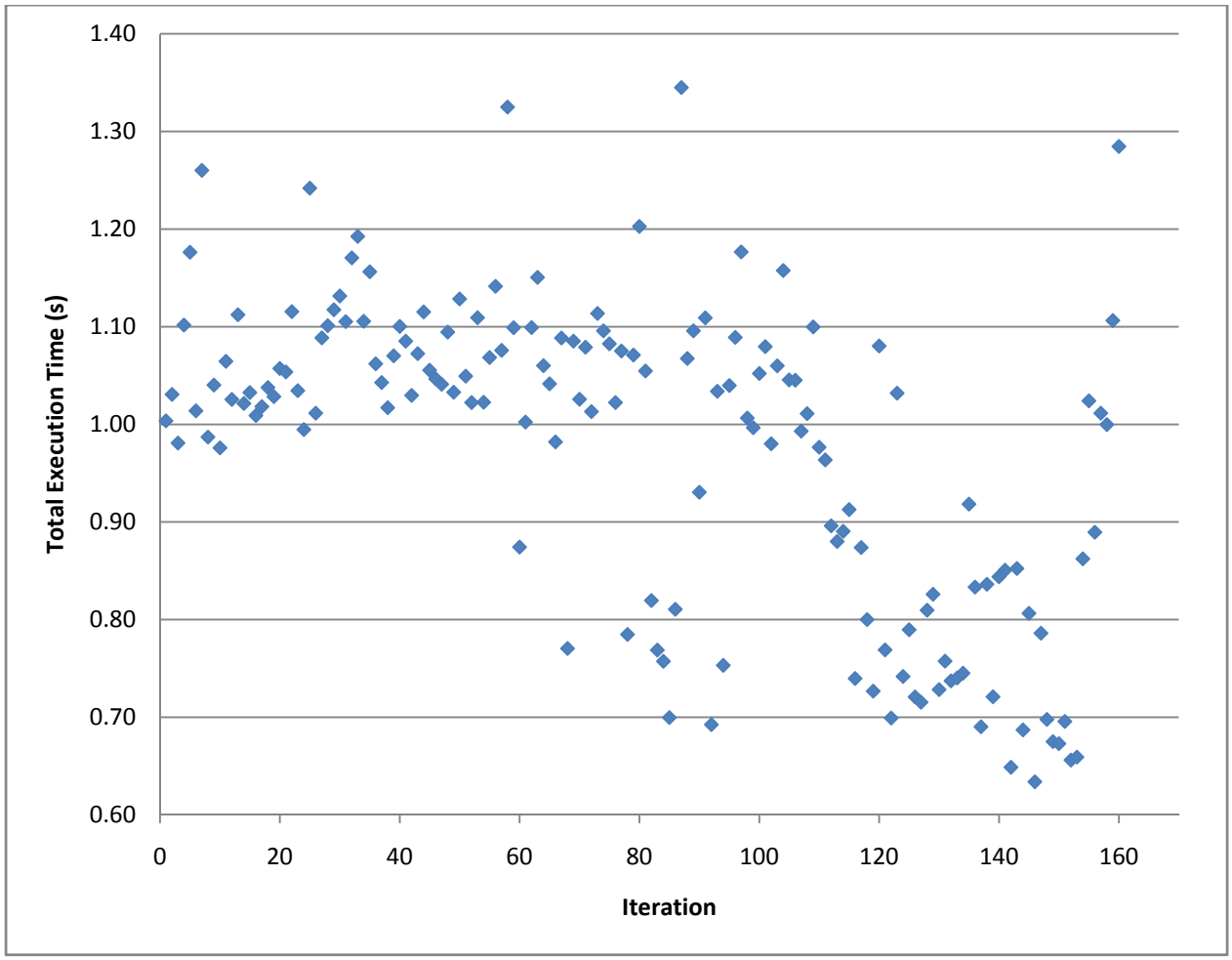



Figure 5-45. Total execution times for the system using the driver side LADAR with a dynamic vehicle and static environment with the presence of sensor noise, the use of position estimation, and access to the world model knowledge store.

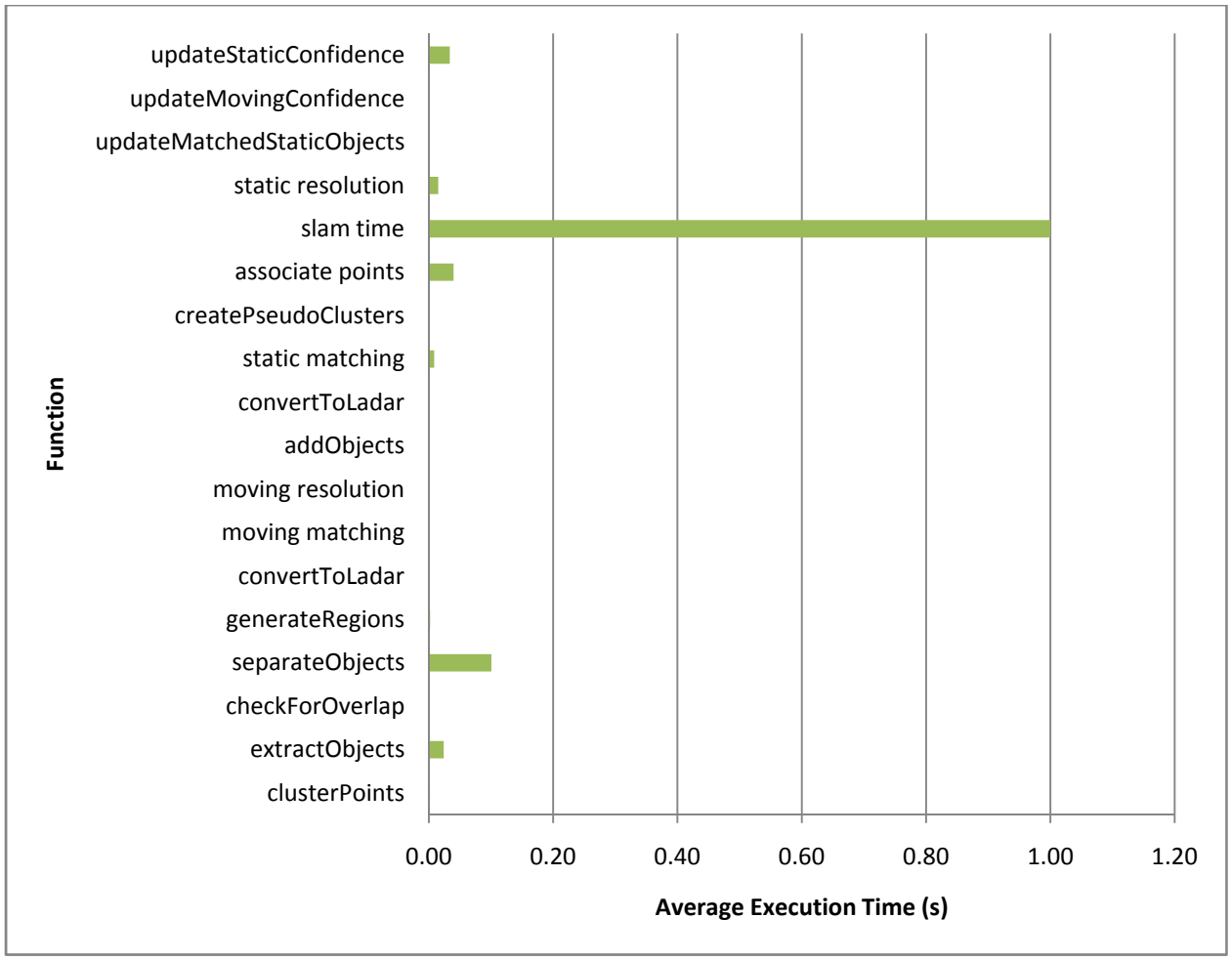



Figure 5-46. Average execution times for different functions using the driver side LADAR with a dynamic vehicle and static environment with the presence of sensor noise, the use of position estimation, and access to the world model knowledge store.

Figure 5-47. Satellite imagery from the Gainesville Raceway with an overlay of LADAR point data from the driver and passenger side LADARs.

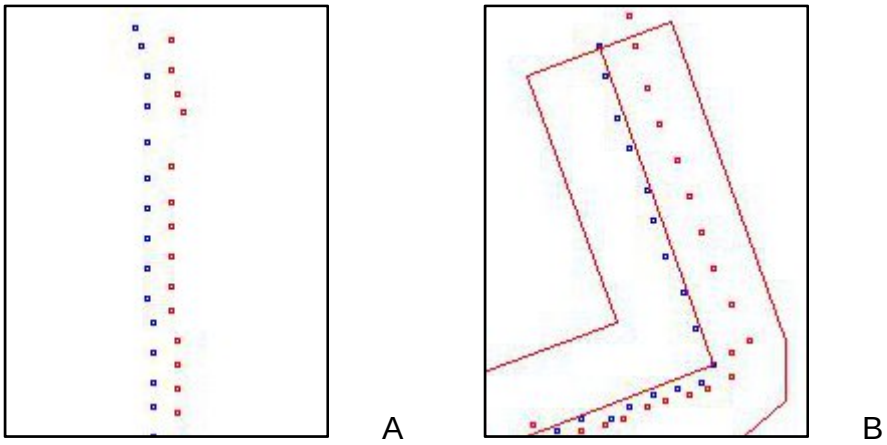



Figure 5-48. Points from the driver and passenger side LADAR aren't aligned. A) The difference in the alignment of the raw data points. B) The difference in the alignment of an object extracted from the passenger side LADAR and the scan points from the driver side LADAR.\

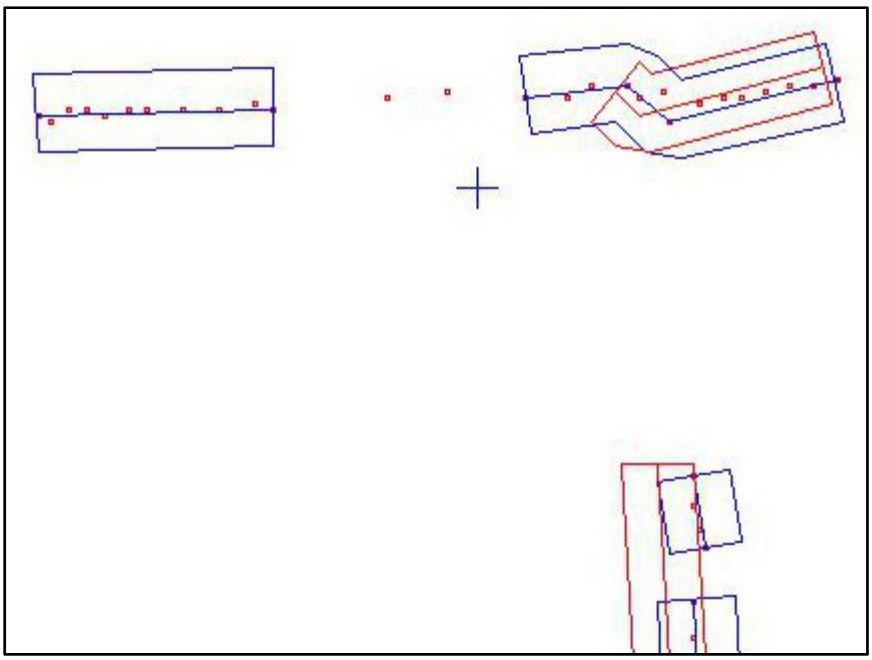


Figure 5-49. Different objects are extracted between the driver and passenger side LADAR.

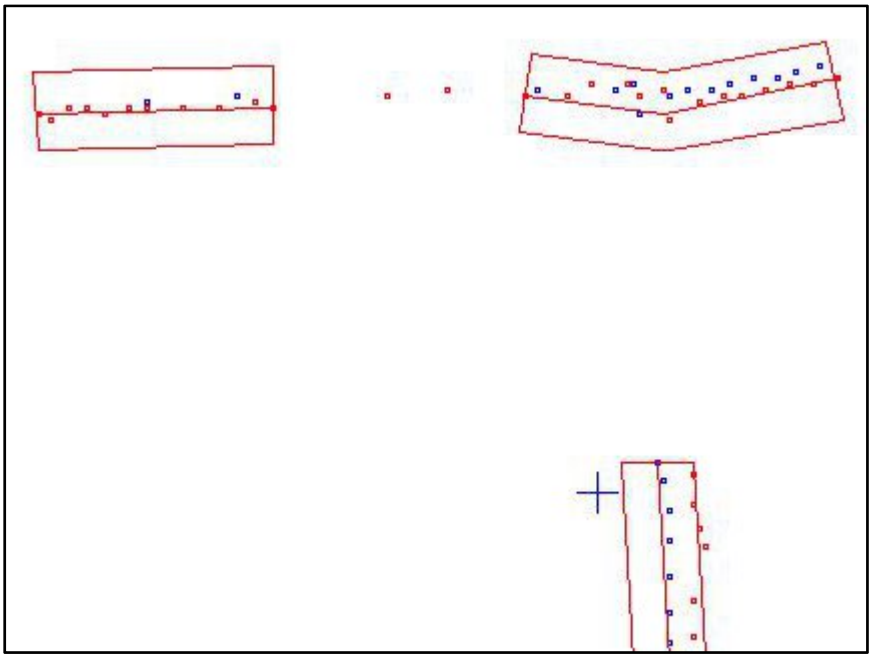
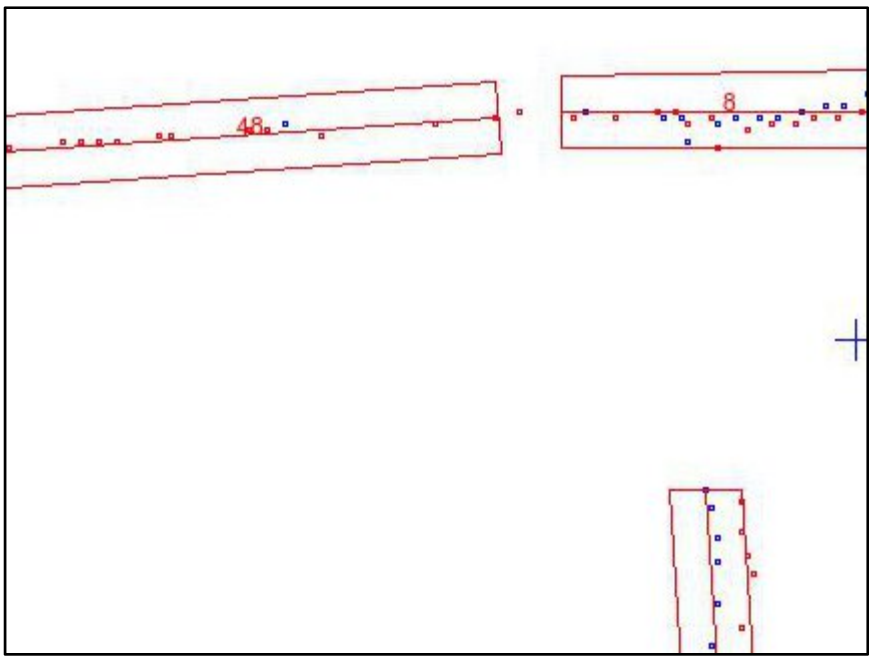
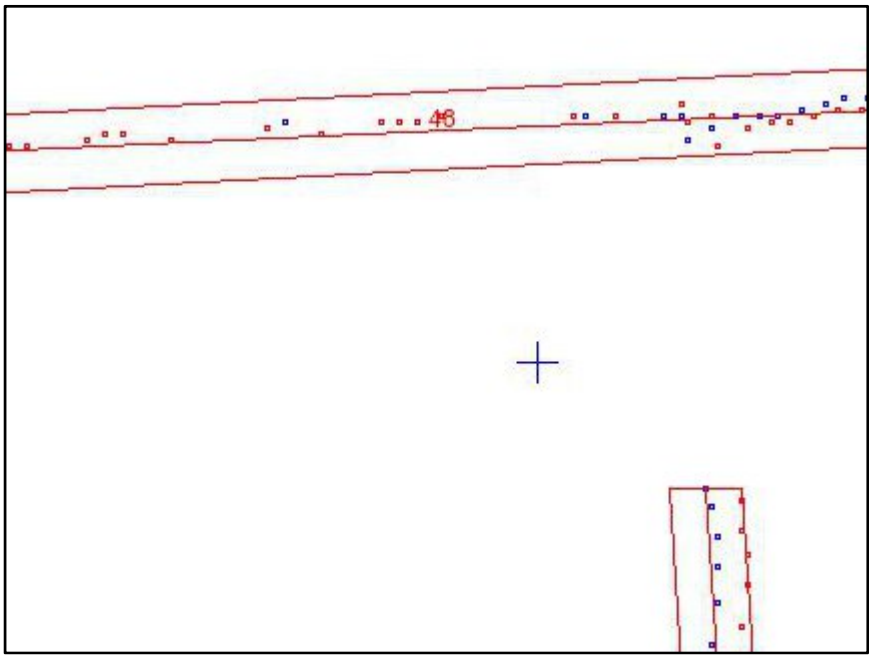


Figure 5-50. Objects are successfully updated by data from the driver and passenger side LADAR.




A




B

Figure 5-51. Objects are updated faster when both LADAR are used.

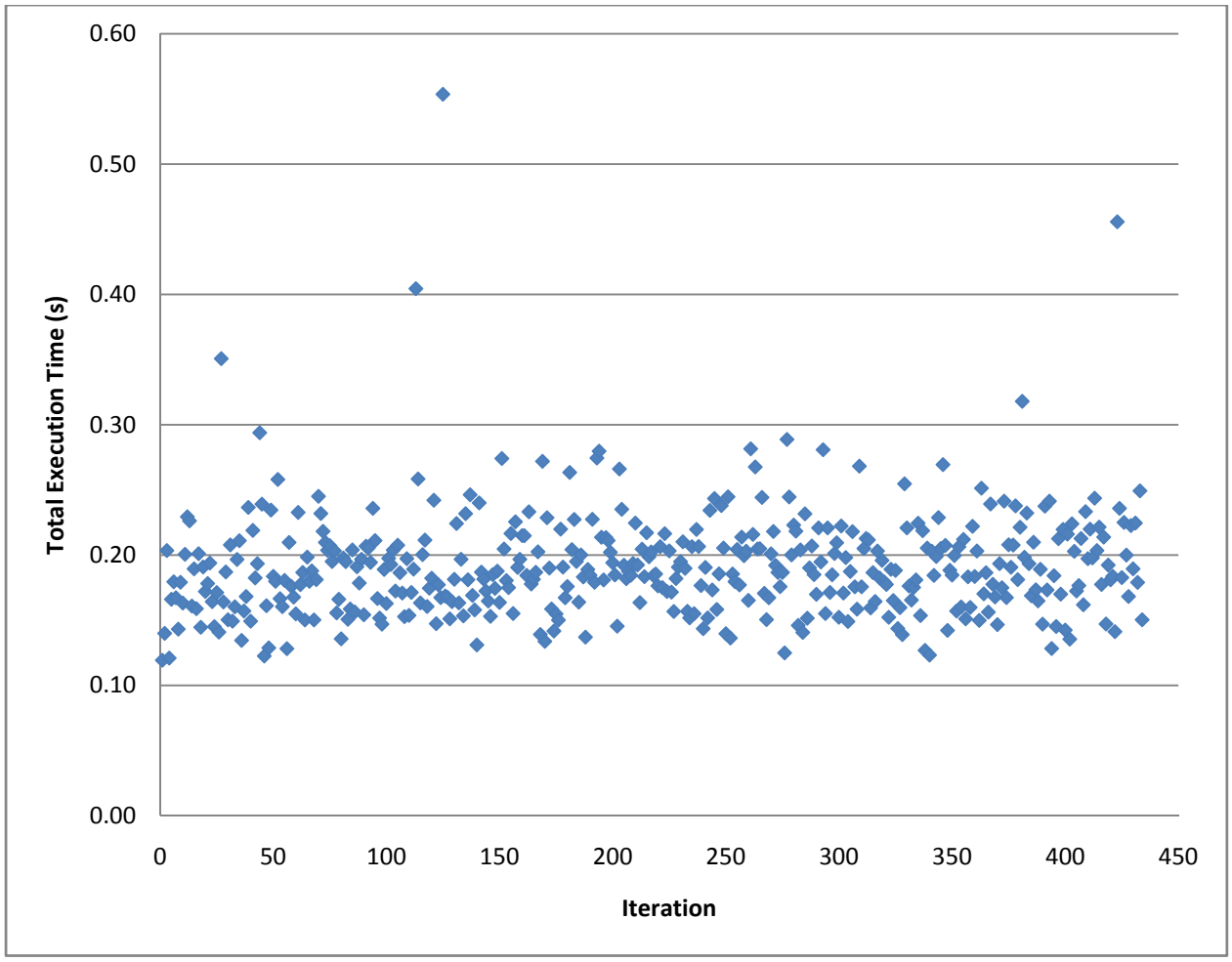



Figure 5-52. Total execution times for the system using both the driver and passenger side LADAR with a static vehicle and static environment with the presence of sensor noise, but without the use of position estimation, or access to the world model knowledge store.

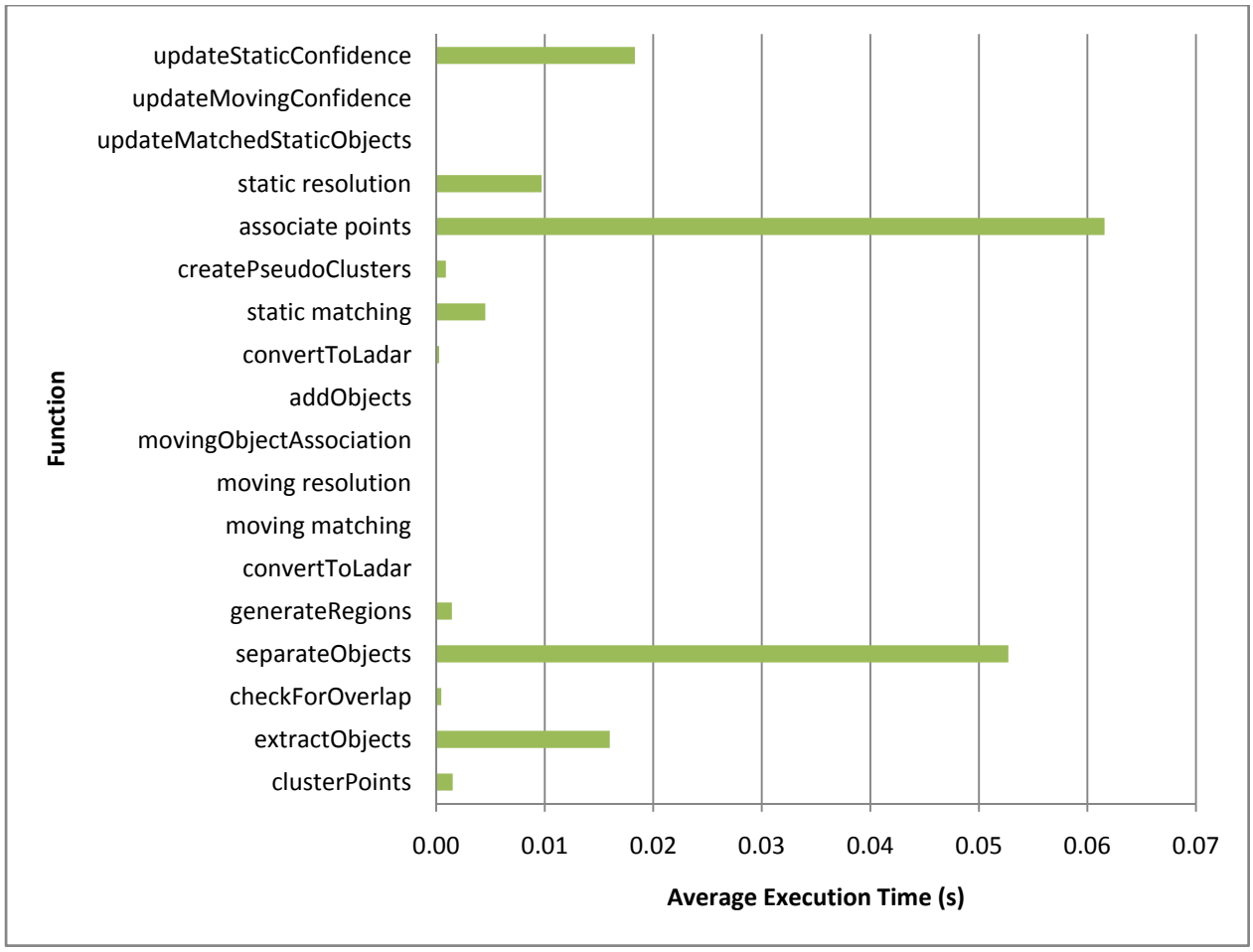
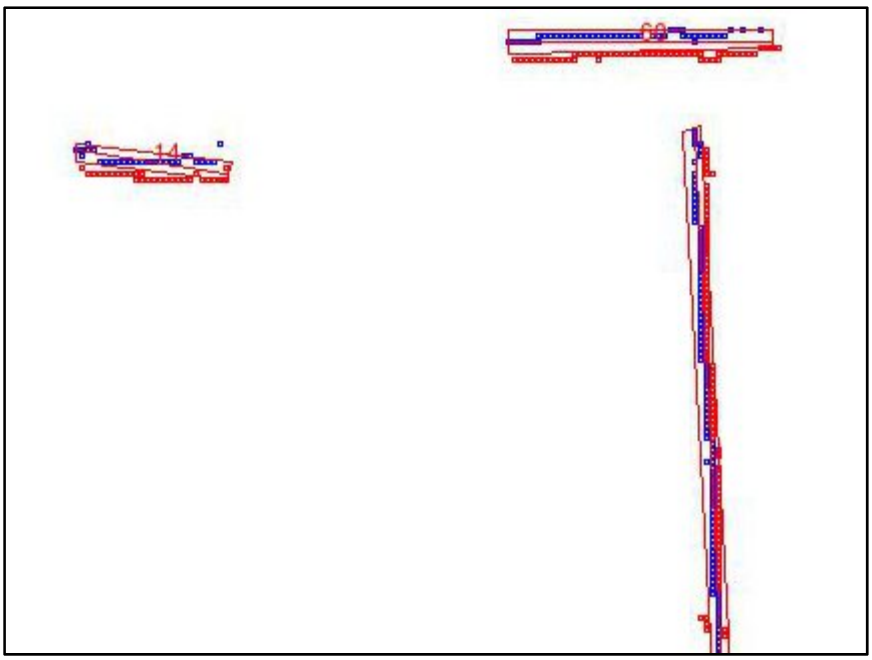
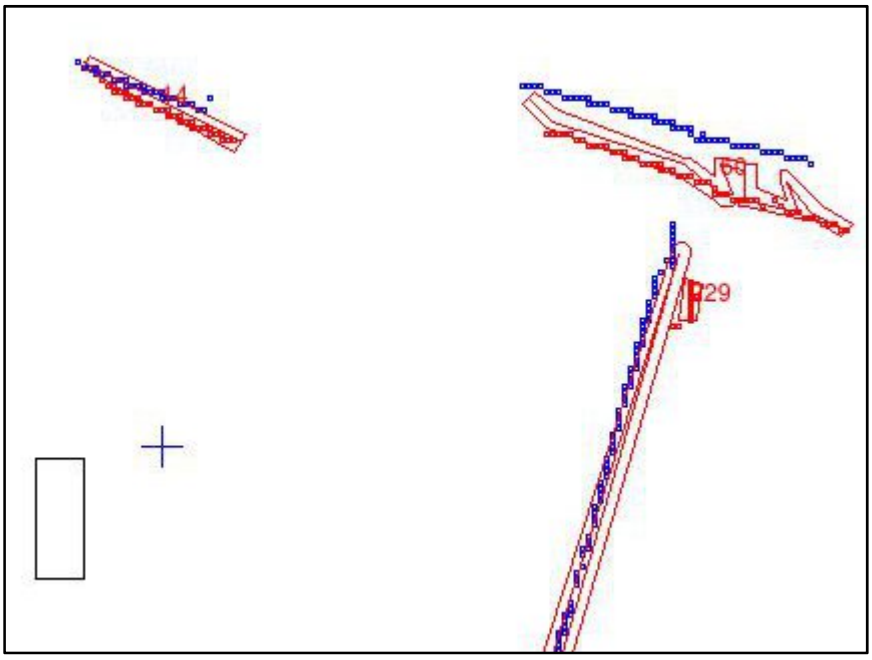



Figure 5-53. Average execution times for different functions using both the driver and passenger side LADAR with a static vehicle and static environment with the presence of sensor noise but without the use of position estimation, or access to the world model knowledge store




A




B

Figure 5-54. Large difference between the driver and passenger side points when platform is in motion. A) The difference between the points when travelling in a straight line. B) The difference between the points when turning.

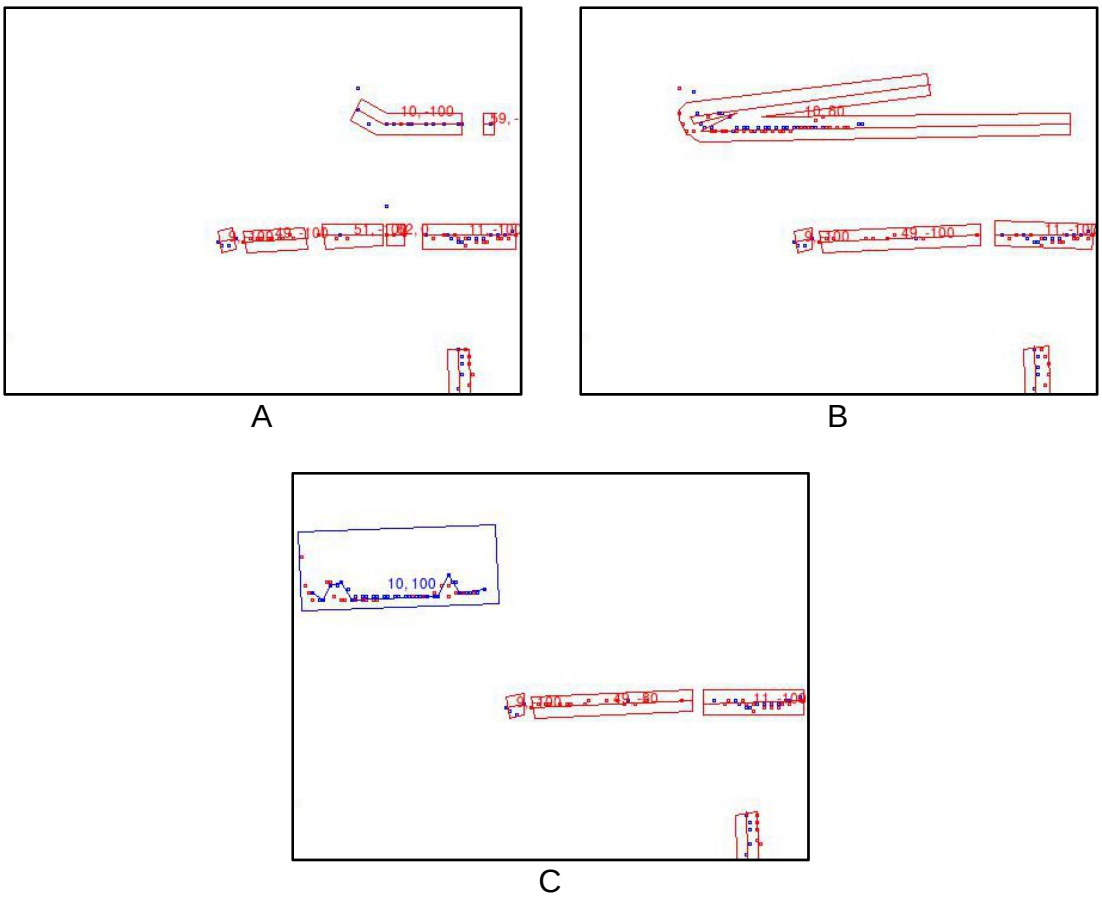
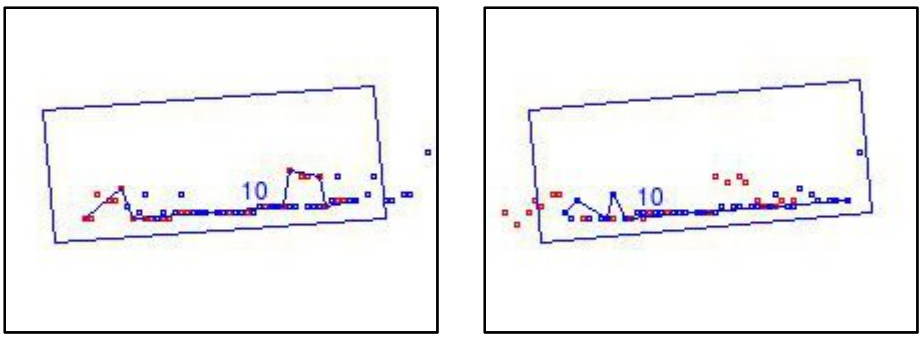



Figure 5-55. Moving object detection using multiple LADAR. A) The object is first detected when is it static. B) The object begins to move but is still misclassified as a static object and is incorrectly updated. C) The object is correctly reclassified as a moving object and the object representation corrected.




A B

Figure 5-56. Placement of moving object changes due to differences between points from each LADAR. A) The object is detected using the driver side points. B) When the object is detected using the passenger side points it appears to move backwards (to the right) despite the fact that the object has moved forward (to the left).

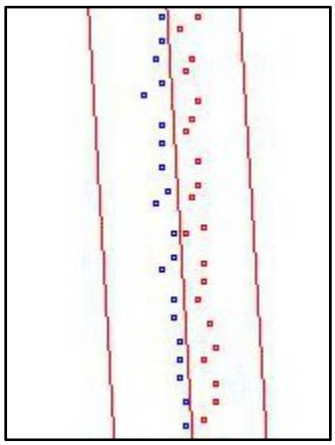


Figure 5-57. The stored object is averaged to lie between the misaligned scan points.

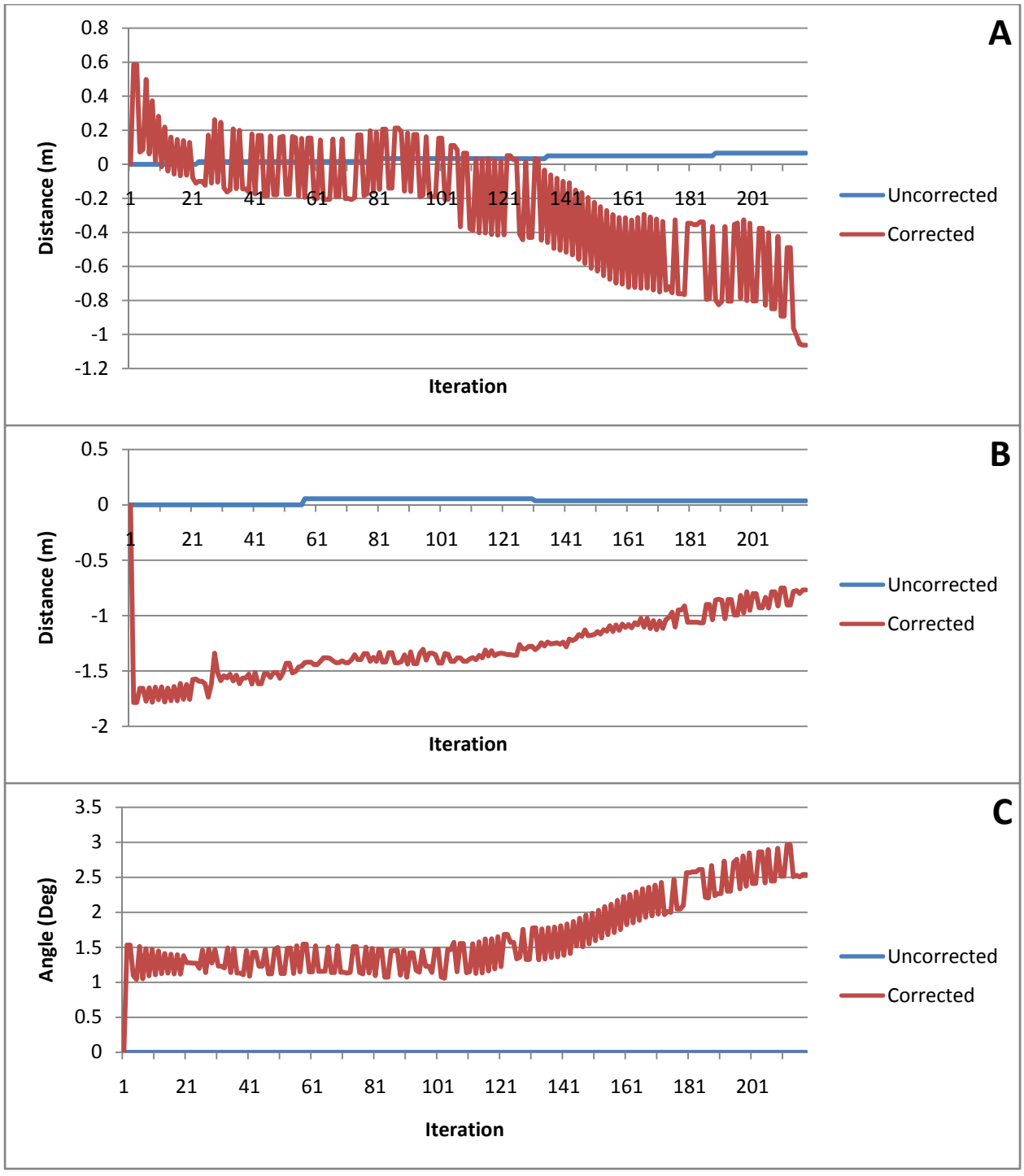



Figure 5-58. Distance from the origin of the corrected and uncorrected position estimates when running with real LADAR data from both the driver and passenger side on a static platform in a static environment. A) Change in the UTM X position. B) Change in the UTM Y position. C) Change in Yaw.

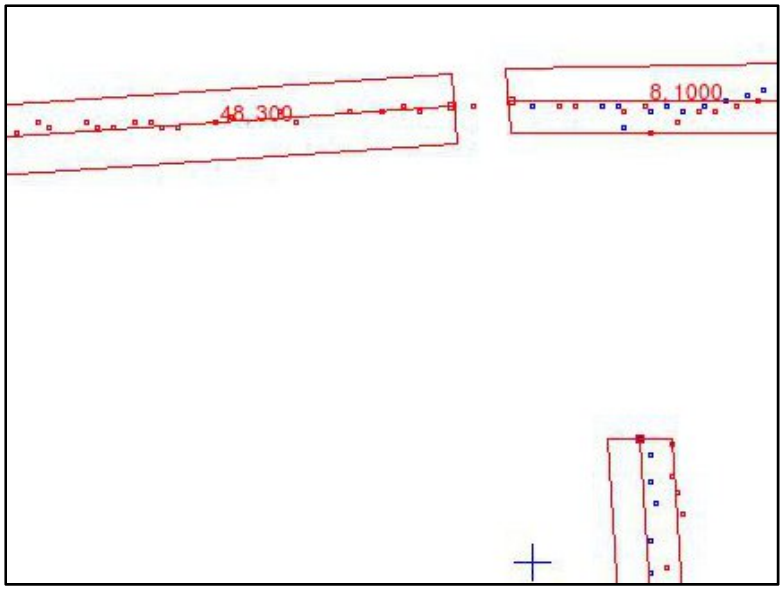
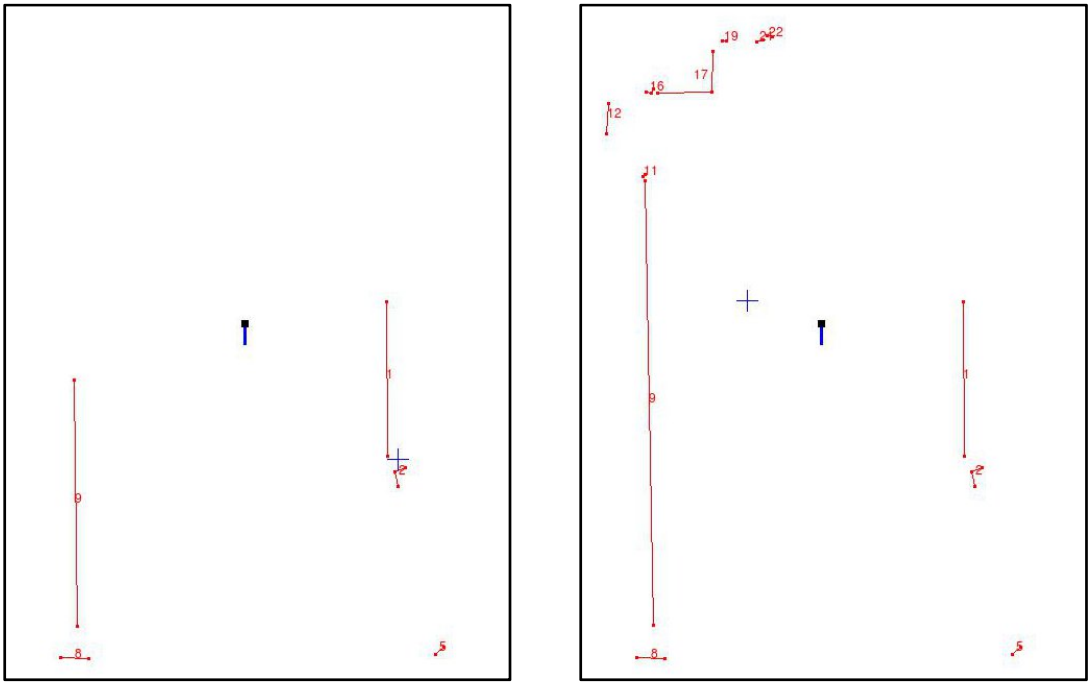
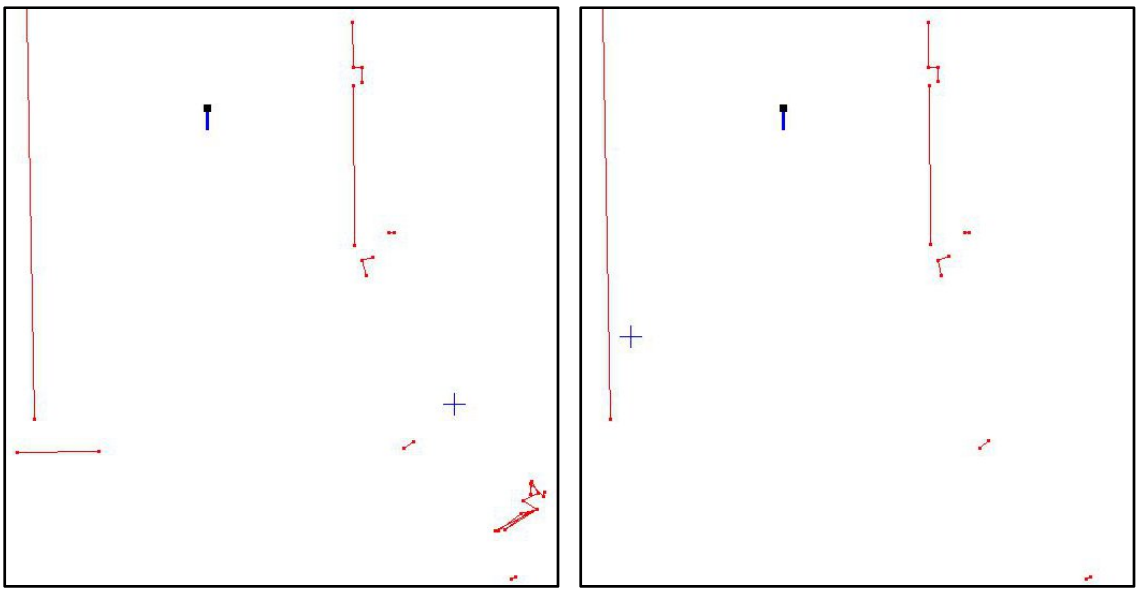



Figure 5-59. Object confidence changes at different rates due to sensor overlap.



A B

Figure 5-60. Objects are added to the WMKS at different times due to the difference rates of change of the object confidence. A) A small number objects are added to the WMKS first. B) More objects object are added to the WMKS after some time.



A B

Figure 5-61. Some detected objects are not added to the WMKS due to discrepancies between the LADAR. A) When using the passenger side LADAR the object detected due to the ground is added to the WMKS. B) When using both LADAR, the object is not added to the WMKS.

# CHAPTER 6
# CONCLUSIONS AND FUTURE WORK

A novel method for performing simultaneous localization, mapping, and moving object tracking has been presented in this dissertation. Also, formalized methodologies for using an external WMKS and fusing data from multiple LADAR were introduced. In this chapter the discussion on the presented research will be finalized by first presenting potential areas of future work. Next, conclusions will be drawn from the discussions on the approach, testing methodology, and results presented in Chapters 3, 4 and 5 respectively. This section also outlines the contributions of the work to the field of robotics.

## Future Work

The research presented here demonstrated the feasibility of a purely feature based approach to SLAM+DATMO, but the system suffered from a number of shortcomings. First, the system used assumptions based on the LADAR characteristics to simplify the line extraction process. One possible improvement would be the use of statistical models to improve the line extraction and object update processes. The iterative end point fit could be augmented through the use of a total least squares fit to more accurately represent the objects in the presence of sensor noise. The object update process would also benefit from the use of statistical methods to improve the object representation when the platform moved through the environment. Another major limitation of the update process is that an object could never be split in two different objects. Consider the case where any alley between two buildings is blocked by a fence due to construction. The two buildings would appear as a single object. However, when

the fence is removed, it would desirable to split the object, which cannot be done with the current algorithm.

When discussing the use of the WMKS many simplifying assumptions were made. One important assumption was that the objects detected by the SLAM+DATMO system were not be modified by any other components on the robot. However, in an autonomous system it is beneficial to use many sensors to improve perception and environmental understanding. Ideally, not only would each sensor would be able to detect and add objects to the WMKS but they would also be able to modify objects added by other components. However, issues with conflict resolution and data synchronization would need to be addressed. If two software components attempt to modify an object but the changes conflict with each other, the question of which component to trust arises. Also, if the asynchronous communication approach described in Chapter 3 is used there will always be some period of time when one of the local caches will be out of sync with the WMKS. The questions on how to handle this problem also need to be answered.

The LADAR fusion scheme introduced was proven to perform reasonably well, but it suffered from error in the LADAR positioning. A formalized methodology for calibrating the multiple LADAR and ensuring they are aligned would be a major improvement. Another interesting topic for future work is in the area of multiple robotic agents. The fusion approach discussed dealt with LADAR on the same platform, but could be feasibly extended for fusing LADAR data from multiple platforms. However, issues dealing with position agreement between agents and transmission time would have to be solved. The scheme was also limited to only allow sequential operations on the

LADAR objects due to problems with conflict resolution and object updating. One area that could be explored is devising a parallel operation methodology for the fusion scheme. The sequential approach outlined is limited on the number of LADAR than can be fused based on the processing speed of the algorithm and the LADAR scan rate. However, this limitation may be overcome through the use of parallel processing.

Finally, the presented work attempted to generate singular representations for detected objects. Although, the system was shown to be reasonably successful there is much room for improvement. Object representations were poor for non-regular objects such as trees and rocks. Also, there was no method for detecting or handling when an object had been completely described. One advantage of the singular representation approach is the ability to add contextual data to the objects. An object could be tagged as a tree, a restaurant, a school, etc. and high-level planner could use the contextual data to modify its plan. Imagine a planner avoiding restaurants downtown in a city at lunch time to avoid the lunch rush, or perhaps redirecting around schools during drop off and pickup times. The ability to use contextual data in planning would be a major step forward for the field of robotics.

## Conclusions

The roles of robotic systems continue to grow daily and are expected to deal with more complex and real-life situations. As the task complexity increases, the need for safety also increases. It is important that autonomous systems be able to perform their jobs without placing themselves or others at risk. The first step towards achieving this goal lies in perception as a robot cannot avoid a situation it cannot sense. This dissertation has presented the author’s work in developing a novel approach for perceiving and understanding the environment around the vehicle in the presence of

moving objects and using multiple LADAR. It also introduced a method for sharing the information with other components within the autonomous system. It began by introducing the challenges associated with perception in autonomous vehicles, especially in the presence of moving objects, and outlined a problem statement in Chapter 1. Next, previous work in the areas of SLAM, DATMO, and SLAM+DATMO were presented and discussed in the Chapter 2. Chapter 3 outlined the author's approach to the problem and details on the implemented method. A description of the testing environment, validation procedure and metrics were provided in Chapter 4 while the obtained results were discussed in Chapter 5.

In general, SLAM and DATMO have been treated separately with little consideration given to the interaction between static and dynamic elements. Also, most SLAM approaches generated either a point or feature map, which had no real-world interpretation. They also did not attempt to detect differences between the map and the sensed environment and simply match the detected points or features for localization purposes. The presented work introduced a method for generating an object map, which has a real-world interpretation and can be enhanced with contextual attributes, such as object classifications, images, etc, using a spatial reconstruction approach. The approach extended and refined an object's representation as the viewing angle changed and more information about the object was known. The map was constructed in the presence of moving objects while simultaneously estimating the vehicle's current position and orientation.

Secondly, the work formally outlined an approach for using a shared world model knowledge store while attempting to maintain real-time sensor processing. Previous,

approaches were self contained and did not share the generated map or detected objects with other elements within the system. Any improvements using additional sensors, such as the addition of cameras, could not be easily integrated and required a major overhaul of the system. As robots develop greater functionality and are tasked with greater responsibilities, modularity becomes very important. It leads to the development of robust vehicles that can function in multiple scenarios and even recover if some of their elements fail. The methodology outlined for using a shared WMKS is an important first step towards that goal.

Finally, an approach for fusing multiple LADAR sources on a single platform was outlined. The use of multiple low-cost LADAR sources to provide a 360 degree view around the vehicle is sometimes preferred to the use of a single, high-cost LADAR especially when there are limitations on sensor placement. However, very few fusion schemes have been formally proposed for dealing with multiple LADAR, especially when keeping the data in vector space. Grid approaches to data fusion either force a loss of data resolution or resort to image processing techniques, which can be slow and complex. Many of the popular vector-based LADAR processing algorithms exploit the sensor properties and break down when those properties are not present. The presented fusion scheme treated each LADAR independently and upheld the sensor property requirements for the vector-based algorithms.

# LIST OF REFERENCES


[1] S. Nof, "HANDBOOK OF INDUSTRIAL ROBOTICS," New York, NY, USA: John Wiley & Sons, 1985.

[2] H.-M. Huang, "Autonomy Levels for Unmanned Systems (ALFUS) Framework Volume I: Terminology Version 1.1," 2004.

[3] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part I," *Robotics & Automation Magazine, IEEE,* vol. 13, pp. 99-110, 2006.

[4] "Urban Challenge Technical Evaluation Criteria," Virginia, 2006.

[5] C.-C. Wang and C. Thorpe, "A hierarchical object based representation for simultaneous localization and mapping," Sendai, Japan, 2004, pp. 412-418.

[6] C. D. Crane III, D. G. Armstrong II, R. Touchton, T. Galluzzo, S. Solanki, J. Lee, D. Kent, M. Ahmed, R. Montane, S. Ridgeway, S. Velat, G. Garcia, M. Griffis, S. Gray, J. Washburn, and G. Routson, "Field report: Team CIMAR's NaviGATOR: An unmanned ground vehicle for the 2005 DARPA Grand Challenge," *Journal of Field Robotics,* vol. 23, pp. 599-623, 2006.

[7] J. Vandorpe, H. Van Brussel, and H. Xu, "Exact dynamic map building for a mobile robot using geometrical primitives produced by a 2D range finder," in *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, 1996, pp. 901-908 vol.1.

[8] B. Schiele and J. L. Crowley, "A comparison of position estimation techniques using occupancy grids," in *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*, 1994, pp. 1628-1634 vol.2.

[9] F. Lu and E. Milios, "Robot pose estimation in unknown environments by matching 2D range scans," *Journal of Intelligent and Robotic Systems: Theory and Applications,* vol. 18, pp. 249-275, 1997.

[10] C. Lara, L. Romero, and F. Calderon, "A robust iterative closest point algorithm with augmented features," Atizapan de Zaragoza, Mexico, 2008, pp. 605-614.

[11] S. T. Pfister, K. L. Kriechbaum, S. I. Roumeliotis, and J. W. Burdick, "Weighted range sensor matching algorithms for mobile robot displacement estimation," Washington, DC, United states, 2002, pp. 1667-1674.

[12] A. Siadat, A. Kaske, S. Klausmann, M. Dufaut, and R. Husson, "An Optimized Segmentation Method For A 2D Laser-Scanner Applied To Mobile Robot Navigation," *IFAC Symposium in Intelligent Components and Instruments for Control Applications,* pp. 153-158, 1997.

[13] V. Nguyen, A. Martinelli, N. Tomatis, and R. Siegwart, "A comparison of line extraction algorithms using 2D laser rangefinder for indoor mobile robotics," in *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, 2005, pp. 1929-1934.

[14] S. Zhang, M. Adams, F. Tang, and L. Xie, "Geometrical Feature Extraction Using 2D Range Scanner," in *Control and Automation, 2003. ICCA '03. Proceedings. 4th International Conference on*, 2003, pp. 901-905.

[15] M. Adams, Z. Sen, and X. Lihua, "Particle filter based outdoor robot localization using natural features extracted from laser scanners," in *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, 2004, pp. 1493-1498 Vol.2.

[16] A. Diosi and L. Kleeman, "Advanced sonar and laser range finder fusion for simultaneous localization and mapping," Sendai, Japan, 2004, pp. 1854-1859.

[17] J. Guivant, E. Nebot, and S. Baiker, "Localization and map building using laser range sensors in outdoor applications," *Journal of Robotic Systems,* vol. 17, pp. 565-583, 2000.

[18] G. A. Borges and M. J. Aldon, "Motion estimation by iterative 2-D features matching in range images," San Francisco, CA, USA, 2000, pp. 3197-3202.

[19] R. Madhavan and H. F. Durrant-Whyte, "2D map-building and localization in outdoor environments," *Journal of Robotic Systems,* vol. 22, pp. 45-63, 2005.

[20] Y. Xu, C. Zhang, W. Bao, L. Su, and M. Wang, "A robust pose estimation algorithm for mobile robot based on clusters," Wuhan, China, 2008, pp. 1003-1010.

[21] S. T. Pfister, S. I. Roumeliotis, and J. W. Burdick, "Weighted line fitting algorithms for mobile robot map building and efficient data representation," Taipei, Taiwan, 2003, pp. 1304-1311.

[22] R. Labayrade, C. Royere, D. Gruyer, and D. Aubert, "Cooperative fusion for multi-obstacles detection with use of stereovision and laser scanner," *Autonomous Robots,* vol. 19, pp. 117-140, 2005.

[23] F. Nashashibi and A. Bargeton, "Laser-based vehicles tracking and classification using occlusion reasoning and confidence estimation," Eindhoven, Netherlands, 2008, pp. 847-852.

[24] E. Prassler, J. Scholz, and A. Elfes, "Tracking multiple moving objects for real-time robot navigation," *Autonomous Robots,* vol. 8, pp. 105-116, 2000.

[25] A. Almeida, J. Almeida, and R. Araujo, "Real-time tracking of moving objects using particle filters," Dubrovnik, Croatia, 2005, pp. 1327-1332.

[26] Y. Jin-Xia, C. Zi-Xing, and D. Zhuo-Hua, "Detection and tracking of moving object with a mobile robot using laser scanner," in *Machine Learning and Cybernetics, 2008 International Conference on*, 2008, pp. 1947-1952.

[27] C. Premebida and U. Nunes, "Segmentation and geometric primitives extraction from 2D laser range data for mobile robot applications," 2005.

[28] R. A. MacLachlan and C. Mertz, "Tracking of moving objects from a moving vehicle using a scanning laser rangefinder," Toronto, ON, Canada, 2006, pp. 301-306.

[29] B. Kluge, C. Kohler, and E. Prassler, "Fast and robust tracking of multiple moving objects with a laser range finder," Seoul, Korea, Republic of, 2001, pp. 1683-1688.

[30] C. Stiller, J. Hipp, C. Rossig, and A. Ewald, "Multisensor obstacle detection and tracking," *Image and Vision Computing,* vol. 18, pp. 389-396, 2000.

[31] K. C. J. Dietmayer, J. Sparbert, and D. Streller, "Model based Object Classification and Object Tracking in Traffic Scenes from Range Images," *Proceedings of IV IEEE Intelligent Vehicles Symposium, Tokyo,* 2001.

[32] D. Streller, K. Dietmayer, and J. Sparbert, "Object tracking in traffic scenes with multi-hypothesis approach using laser range images," in *8th World Congress on Intelligent Transport Systems*, Sydney, Australia, 2001.

[33] D. Streller and K. Dietmayer, "Object tracking and classification using a multiple hypothesis approach," Parma, Italy, 2004, pp. 808-812.

[34] A. Mendes, L. C. Bento, and U. Nunes, "Multi-target detection and tracking with a laserscanner," Parma, Italy, 2004, pp. 796-801.

[35] G. A. Borges and M.-J. Aldon, "Line extraction in 2D range images for mobile robotics," *Journal of Intelligent and Robotic Systems: Theory and Applications,* vol. 40, pp. 267-297, 2004.

[36] C. Premebida and U. Nunes, "A multi-target tracking and GMM-classifier for intelligent vehicles," Toronto, ON, Canada, 2006, pp. 313-318.

[37] C. Premebida, G. Monteiro, U. Nunes, and P. Peixoto, "A Lidar and vision-based approach for pedestrian and vehicle detection and tracking," Seattle, WA, United states, 2007, pp. 1044-1049.

[38] M. Lindstrom and J. O. Eklundh, "Detecting and tracking moving objects from a mobile platform using a laser range scanner," Maui, HI, United states, 2001, pp. 1364-1369.

[39] F. Fayad and V. Cherfaoui, "Tracking objects using a laser scanner in driving situation based on modeling target shape," Istanbul, Turkey, 2007, pp. 44-49.

[40] H. Zhao, X. W. Shao, K. Katabira, and R. Shibasaki, "Joint Tracking and Classification of Moving Objects at Intersection Using a Single-Row Laser Range Scanner," in *Intelligent Transportation Systems Conference, 2006. ITSC '06. IEEE*, 2006, pp. 287-294.

[41] C.-C. Wang, C. Thorpe, S. Thrun, M. Hebert, and H. Durrant-Whyte, "Simultaneous localization, mapping and moving object tracking," *International Journal of Robotics Research,* vol. 26, pp. 889-916, 2007.

[42] T.-D. Vu, O. Aycard, and N. Appenrodt, "Online localization and mapping with moving object tracking in dynamic outdoor environments," Istanbul, Turkey, 2007, pp. 190-195.

[43] C.-C. Wang and C. Thorpe, "Simultaneous localization and mapping with detection and tracking of moving objects," Washington, DC, United states, 2002, pp. 2918-2924.

[44] H. Zhao, M. Chiba, R. Shibasaki, X. Shao, J. Cui, and H. Zha, "A laser-scanner-based approach toward driving safety and traffic data collection," *IEEE Transactions on Intelligent Transportation Systems,* vol. 10, pp. 534-546, 2009.

[45] N. Johnson, C. D. Crane, III, A. Arroyo, and E. Schwartz, "Feature-Based Object Detection using Multiple 2-D Laser Range Finders," in *Florida Conference on Recent Advances in Robotics* Jacksonville, FL, 2010.

[46] D. Kent, "Storing and Predicting Dynamic Attributes in a World Model Knowledge Store," in *Department of Mechanical and Aerospace Engineering*. vol. Ph.D. Gainesville, FL: University of Florida, 2007.

# BIOGRAPHICAL SKETCH

Nicholas McKinley Johnson was born and raised in the twin island nation of Trinidad and Tobago, where he attended St. Mary's College, one of the top high schools in the country, and graduated near the top of his class in 1999. In 2001, he was awarded a Founders Scholarship to attended Howard University in Washington, DC. During his time at Howard he was inducted into the Golden Key Honor Society and Tau Beta Pi: The Engineering Honor Society, and also served as the President of the Billiards club, the Vice-President of the Robotics Club, and the Cataloger of the student chapter of Tau Beta Pi. Nicholas was also awarded the International Engineering Consortium William L. Everitt Student Award of Excellence and received first place for his senior project at the 15$^{th}$ Annual Electrical and Computer Engineering day. He received a Bachelors of Science in electrical engineering in 2005 and graduated second in his class with Summa Cum Laude honors. He was accepted to the PhD program at the University of Florida in Gainesville, Fl where he received his Masters of Science in electrical engineering in August 2010 and his PhD in December 2010. During his time at Florida he worked at the Center for Intelligent Machines and Robotics and was one of the lead members for University of Florida's 2007 Defense Advanced Research Projects Agency Urban Challenge team. His research focused on perception in autonomous ground vehicles but he is also interested in world modeling and artificial intelligence.